# A Systematic Approach to Hazard and Operability Study (HAZOP)

# A Systematic Approach to Hazard and Operability Study (HAZOP)

By

Paul Aoanan, B.Eng. & Mgmt. Mechatronics

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfillment of the Requirements for the Degree

Master of Applied Science

McMaster University

Master of Applied Science (April, 2021) McMaster University
(Software Engineering) Hamilton, Ontario

| | |
|---|---|
| Title: | A Systematic Approach to Hazard and Operability Study (HAZOP) |
| Author: | Paul Aoanan, B.Eng. & Mgmt. Mechatronics (McMaster University) |
| Supervisors: | Alan Wassyng, Mark Lawford |
| Number of Pages: | x, 97 |

# Abstract

A system safety assurance case aims to demonstrate that a system is reasonably safe within the parameters defined according to its intended use. A system safety assurance case involves the definition of a Safety Engineering Process and its execution for the particular system. An essential element in the Safety Engineering Process is *hazard analysis*. An often used version of hazard analysis is *HAZOP*. HAZOP identifies hazards and hazardous events in the system's design. Traditionally, HAZOP is performed based on the expertise of a multi-disciplinary team. This team uses a heuristic based approach that results in documented output that often does not include adequate traceability as to how the output results were obtained. This thesis proposes a systematic approach to HAZOP that was developed after performing detailed analysis on how traditional HAZOP is performed in industry. It aims to produce documented output in which the output results are traceable to interim steps in the process. We call this systematic approach HAZOP$^+$, because it was designed to provide sufficient detail so that it can form the basis of a HAZOP metamodel created in Workflow$^+$ – a relatively new model driven methodology for developing assurance cases. Workflow$^+$ has well-defined semantics, and so we refer to HAZOP$^+$ as *formalizable*. HAZOP$^+$ has a number of benefits over traditional HAZOP, and these benefits are demonstrated by comparing

a traditional application of HAZOP with the application of HAZOP$^{+}$, both applied to a typical *Lane Keeping Assist* feature. A long term objective of system safety assurance is to be able to perform *incremental safety assurance*, for example, by updating the system safety assurance case after a modification to the system or its environment. Since the safety assurance case for a system depends on elements of the Safety Engineering Process, as well as the outputs of that process, the ability to perform an incremental hazard analysis after a modification to the system or environment can be a real benefit. This thesis further describes how HAZOP$^{+}$ can be enhanced/extended to HAZOP$^{\Delta}$ – an incremental version of HAZOP$^{+}$.

# Acknowledgements

First and foremost, I would like to convey my deepest gratitude and respect to my thesis supervisors, Dr. Alan Wassyng and Dr. Mark Lawford. They have inspired me to become a well-rounded professional and helped me realize the power of higher education. They have demonstrated what hard work, grit, and perseverance can accomplish.

I would also like to thank Dr. Richard Paige and Dr. Spencer Smith for their willingness to serve on the Defence Committee and for providing valuable feedback.

My sincere appreciation must also go to the member of my thesis advisory committee, Dr. Zinovy Diskin. He generously gave his time to offer valuable comments while accommodating my hectic work-life schedule.

Thank you to Joseph D'Ambrosio, Galen Ressler, Sigrid Wagner, Sahar Kokaly, Lucian Patcas, and Ramesh S for providing invaluable advice and feedback on my work from an industry perspective.

I am grateful to the leadership team at my place of work: Nigel Fonseca, David Tremaine, Ryan Simpson, and Ed Mischkot who have been supportive throughout this incredible journey.

I would like to thank my future family-in-law for their concern for my overall well-being and for teaching me the importance of furthering my interests.

I would like to express my deepest gratitude to my family, Blesilda Aoanan, Danilo Aoanan, and Anthony Aoanan for their unconditional trust, endless patience, and continuous encouragement.

Finally, I would like to provide a most sincere and heartfelt thank you to my partner, Hazel Santos. It was her heart-warming kindness, unfailing support, dependable companionship, and endless encouragement that has provided a new meaning to one's capacity for greatness.

# Contents

# List of Figures

# List of Acronyms

| Acronym | Definition |
|---|---|
| BNT | Badness Network Transfer |
| BTA | Badness Transfer Analysis |
| CTCS | Chinese Train Control System |
| ECU | Engine Control Unit |
| EPS | Electronic Power Steering |
| FA | Functional Architecture |
| FAD | Functional Architecture Diagram |
| FFA | Function Failure Analysis |
| FPTC | Fault Propagation and Transformation Calculus |
| FPTN | Fault Propagation and Transformation Notation |
| GSN | Goal Structuring Notation |
| HAZOP | Hazard and Operability Study |
| IEC | International Electrotechnical Commission |
| ISO | International Organization for Standardization |
| I/O (IO) | Input/s and Output/s |
| LKA | Lane Keeping Assist |
| PHA | Preliminary Hazard Analysis |
| SEP | Safety Engineering Process |
| UML | Unified Modelling Language |
| V&V | Verification and Validation |

# Declaration of Academic Achievement

My contribution has been to the development of the systematic and formal approach to HAZOP. Leveraging my experience in the nuclear and power generation industry in this study, this proposed approach aims to satisfy the definition and needs of HAZOP according to a safety standard such as ISO 26262. This involved thorough investigation and analysis of a typical Safety Engineering Process (SEP) practised in industry and a sample instance of HAZOP at the Preliminary Hazard Analysis (PHA) level. I performed gap analysis and developed the systematic and formalizable HAZOP$^+$ process that addresses the identified gaps. I applied the proposed HAZOP$^+$ to an instance of design data and compared against the industry's typical approach. This was then documented and applied to a major technical report for a proprietary research project. This has facilitated in tailoring HAZOP$^+$ to extend to an incremental HAZOP$^+$ to be usable at any level of abstraction and to be applicable to other stages of the SEP other than the PHA. Furthermore, the proposed incremental HAZOP$^+$ supports development of the SEP towards incremental system safety assurance cases.

# Chapter 1

# Introduction

## 1.1 Motivation

In the last decade, a paradigm shift has occurred in the automotive industry towards developing fully autonomous road vehicles[1]. Decision-making has been delegated to the embedded software that controls the vehicle's electro-mechanical systems. Modern non-autonomous vehicles now use software to control many aspects of the vehicle's behaviour, including semi-autonomous features. All of these vehicles are classified as safety critical systems in which a failure can cause death, injury, financial loss, and environmental damage [2]. To address the functional safety aspect of such systems specifically within road vehicles, the ISO 26262 series of standards has been created in adaptation of the IEC 61508 series of standards [3]. The scope of the adaptation applies to all activities performed during the safety lifecycle of safety-related systems comprised of electrical, electronic, and software components [3]. The standards emphasize the need to provide evidence that functional safety objectives are satisfied, documented, and verified via formal methods [3, 4].

*Assurance Cases* are generalizations of *Safety Cases*, which have been in use for decades, especially in the U.K. An assurance case provides an explicit and convincing argument as to why a system satisfies specific properties [5]. In addition, an assurance case can be used to show compliance with the requirements of ISO 26262 and thus document system safety assurance [2]. An assurance case dedicated to argue the safety properties of the system is called a *Safety Case*. Incremental safety assurance is generally difficult since safety is a global property of the system, and changes in the system or assurance case itself may propagate in ways that are difficult to predict. Thus, although incremental design is common in most industry domains, incremental safety assurance is incredibly challenging to achieve. According to Chowdhury *et al*, *"An assurance case template is a complete assurance case for a product-line, developed prior to building products of that product-line"* [2]. Assurance case templates can serve as the basis for developing assurance cases for a variety of closely related products. Thus, an assurance case template can facilitate incremental assurance.

Iterative modifications made to the design are subject to impact and traceability analyses, which further affect verification and validation efforts. This results in an increased need for modularity in design and development. To further promote objective analyses in the incremental software safety lifecycle, the design and evidence that satisfy safety-related requirements must be explicit, precise, documentable, and reliable.

ISO 26262-1 defines the following terms:

- *Item* as "a system or combination of systems to which ISO 26262 is applied, that implements a function or part of a function at the vehicle

level",

- *Hazard* as a "potential source of harm caused by malfunctioning behaviour of the item",
- *Operational Situation* as a "scenario that can occur during a vehicle's life", and
- *Hazardous Event* as a "combination of a hazard and an operational situation" [14].

According to Clause 7 of ISO 26262-3, *"The functional safety requirements shall specify, if applicable, strategies for... avoidance or mitigation of a hazardous event due to improper arbitration of multiple control requests generated simultaneously by different functions"* [3].

ISO 26262 requires that part of the safety analysis process is to develop a complete set of effective functional safety requirements [3]. One of the analysis methods used is Hazard and Operability Study (HAZOP) to identify hazards and hazardous events. From these, functional safety requirements are developed to mitigate the hazards and hazardous events identified. HAZOP can be performed at any level of abstraction (system to item level) and at any point in the *Safety Engineering Process (SEP)* as the design gets more defined and detailed.

Any change in the design basis, either by design modification, or as part of incremental development, can undergo HAZOP to identify changes in the set of hazards and hazardous events. These changes will then be propagated to the set of functional safety requirements. A formalizable HAZOP process can be applied and propagated to any level of abstraction and SEP; such is a

consequence of formalization.

The author works as a consultant specializing in the development and assurance of software intensive safety critical systems. As a consultant he is keenly aware of the challenges involved in achieving incremental safety assurance. As an important first step, this thesis defines and develops a formalizable HAZOP process that can be extended in ways that enable it to be used efficiently in development processes that take advantage of incremental design. This will greatly help the cause in developing incremental system safety assurance case templates.

## 1.2 HAZOP$^+$ and Workflow$^+$

Workflow$^+$is a model-based framework with precisely defined semantics in which arguments or evidence can be precisely represented [6]. Workflow$^+$models process, product, environment, and the resulting assurance case with extensive traceability. It is less ad hoc than current assurance case development methods. A formalizable HAZOP definition is the basis for a process that can be accurately and precisely modelled in Workflow$^+$. A Workflow$^+$ model of HAZOP$^+$supports the case for incremental hazard analysis and the development of incremental system safety assurance. The work in this thesis has led to the development of Workflow$^+$ models of HAZOP$^+$ in Figure 1.1, Figure 1.2, Figure 1.3, and Figure 1.4.

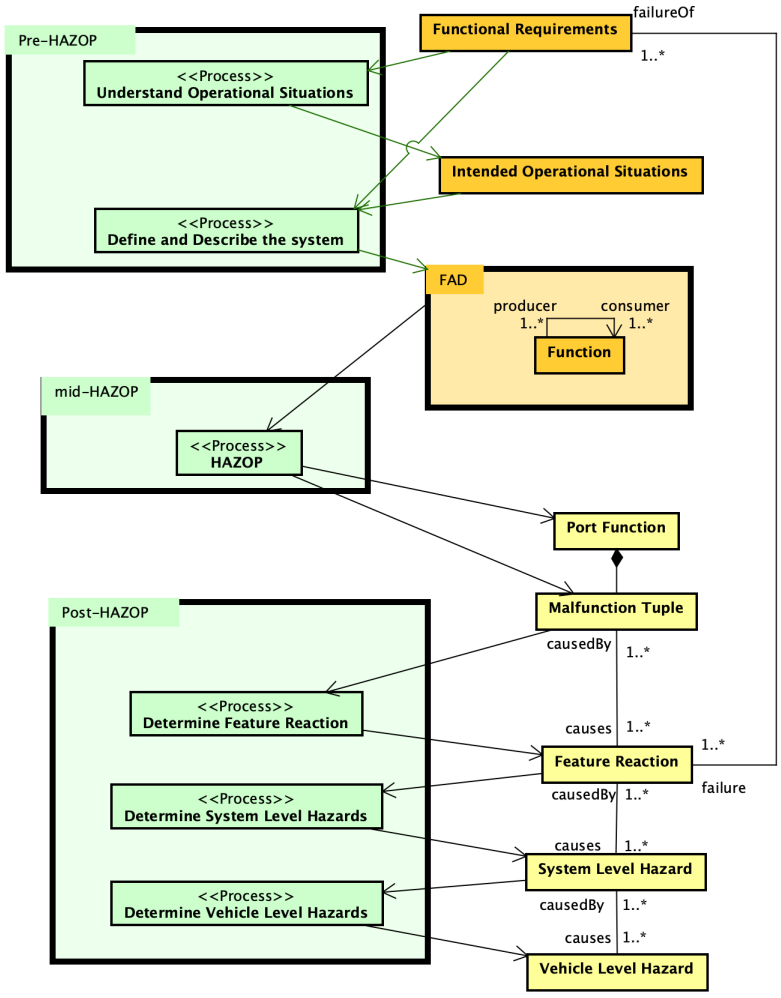



Figure 1.1: HAZOP$^+$ Model in Workflow$^+$

## 1.3 Thesis Goals

The primary goal of this thesis is to propose a systematic approach to a HAZOP process framework that can be modelled in Workflow$^+$. The proposed HAZOP process exhibits the characteristics listed below, and aims to support the development of incremental assurance cases.

- Formalizable

  We use a formal syntax and notation to describe our HAZOP process,

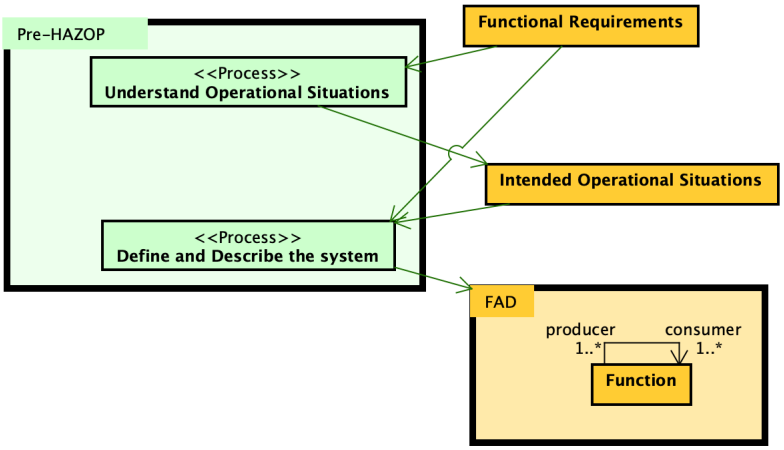



Figure 1.2: Refined Pre-HAZOP$^+$ Model in Workflow$^+$

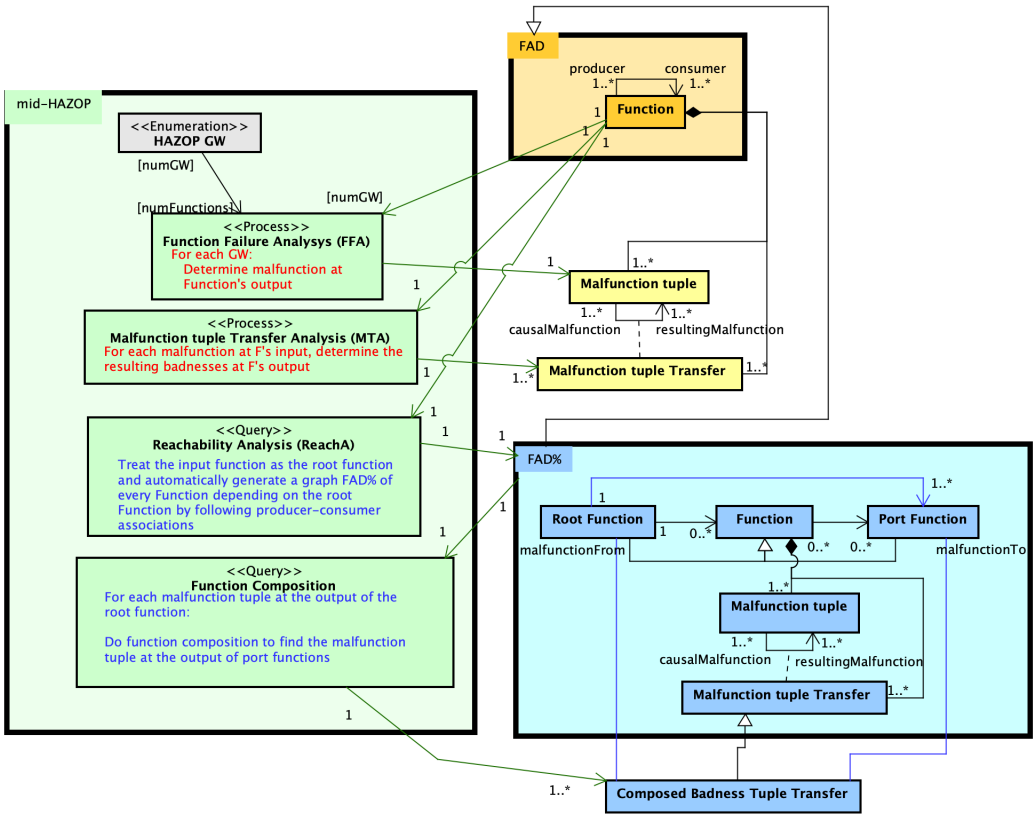



Figure 1.3: Refined Mid-HAZOP$^+$ Model in Workflow$^+$

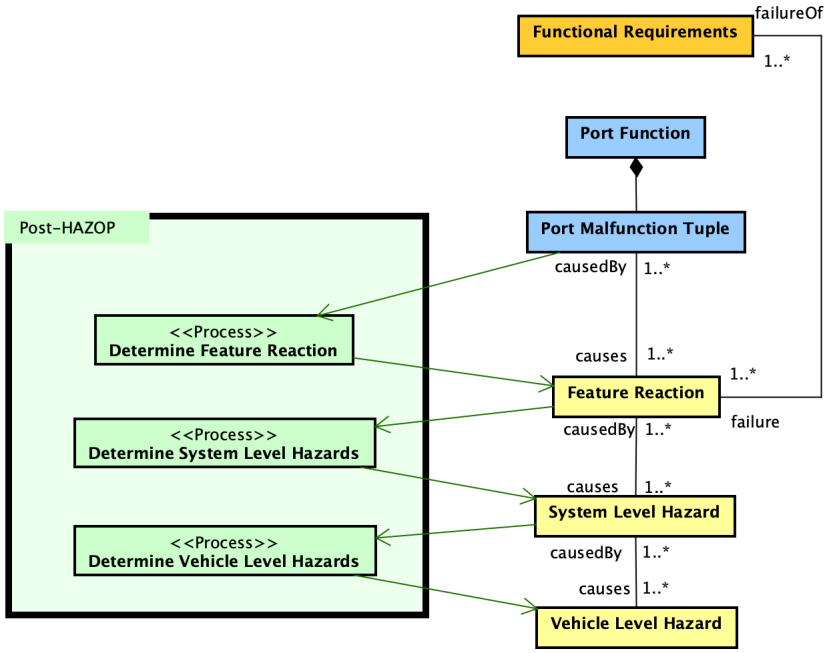



Figure 1.4: Refined Post-HAZOP$^+$ Model in Workflow$^+$

which we call HAZOP$^+$. The process is rigorously and semi-formally defined to a level of detail that facilitated a fully formal version produced subsequent to the work described in this thesis.

- Traceable

  The proposed incremental HAZOP$^+$ takes advantage of the use of *Functional Architecture Diagram (FAD)*, *Goal Structuring Notation (GSN)* -inspired diagrams, and *Requirement Decomposition (RD)* which are inherently hierarchical and traceable as the schemas and semantics can be formally defined.

- Documentable

  The proposed HAZOP$^+$ defines a systematic and formalizable approach to the analysis. This defines a structure not only for the process outputs,

but also for the intermediate analysis steps through the use of architecture diagrams and table-based failure and failure-propagation analysis to determine system level failures.

- Explicit

  As the structure is defined in all aspects of the proposed HAZOP$^+$ , HAZOP analysis and its results can be performed and documented explicitly.

- Verifiable

  Verifiability requires deterministic characteristics and traceability. The defined characteristics in the above bulleted points, should make it possible to verify the proposed HAZOP$^+$ 's process and outputs.

To address the requirement that the design and evidence that satisfy safety-related requirements shall be formal, modular, explicit, precise, documentable, and reliable, this thesis aims to enforce the use of FADs, GSN-inspired diagrams, and tabular expressions in performing and documenting the HAZOP process and its products.

## 1.4 Contribution

The author of this thesis played a major role in the research effort in the development of a systematic and formal HAZOP process based on the work in this thesis. This thesis documents the development of HAZOP$^+$and was used in preparation of a proprietary technical report [7] as part of a research project in the McMaster Centre for Software Certification (McSCert) in developing formal safety assurance.

In detail, the author of this thesis contributed the following to the research effort:

- Performed research into details of general industry approaches to HAZOP,
- Performed requirements decomposition over functional models as the basis for HAZOP$^+$,
- Developed the high-level workflow for HAZOP$^+$, and
- *At the Preliminary Hazard Analysis* (PHA) level:
  - Performed HAZOP$^+$ on an instance of *Lane Keeping Assist (LKA)*,
  - Developed the requirements hierarchy from Requirements Decomposition presented via GSN-inspired diagrams,
  - Developed functional models of the Car-Driver-Environment-Feature that correspond to the functional decomposition process, and
  - Included HAZOP analysis based on the decomposed functions as an integral part of HAZOP$^+$.

## 1.5 Notes on Notation

**Boundary Functions**

The FAD and models involving control theory presented in this thesis make use of the concept of Input and Output Boundary Functions. These are defined in Chapters 5.2 and 5.4. The Boundary Functions are located at the border of a function/feature block overlapping the edges. The Boundary Functions do

not denote the use of ports and are not to be confused as such; it is contrary to that of the Unified Modelling Language (UML).

**Figure and Table Colours**

The figures and tables contained in this thesis incorporate the use of colours with the sole purpose of illustrating the different sections of analysis and levels of hierarchy.

## 1.6 Contents

This subsection discusses the contents of this thesis. Chapter 2 provides an in-depth explanation of how traditional HAZOP is performed typically in industry. Chapter 3 provides relative insight as to how others in the scientific community adopt and apply the HAZOP process. Chapter 4 discusses questions arising from typical applications of HAZOP in industry.

Before the proposed systematic HAZOP$^+$ can be introduced, certain concepts need to be discussed for consistency. Chapter 5 defines the preliminary concepts and definitions needed for HAZOP$^+$.

Next, Chapter 6 defines HAZOP$^+$, the proposed systematic approach to HAZOP, at the Preliminary Hazard Analysis (PHA) level of the Safety Engineering Process (SEP). Chapter 7 demonstrates an instance of executing the proposed HAZOP$^+$ approach on the Lane Keeping Assist feature at the PHA level.

Chapter 8 provides an introduction and execution of the incremental HAZOP$^+$ (a formally defined HAZOP process outside the scope of this thesis) which ties the proposed HAZOP$^+$ approach with requirement decomposition performed over GSN-inspired diagrams and FAD.

Chapter 9 summarizes the contents, analyzes the findings of this thesis, and

discusses the benefits of the proposed HAZOP$^+$. Chapter 10 provides insight on future work and Chapter 11 concludes the thesis.

# Chapter 2

# A Sample Instance of Industry Hazard and Operability Study (HAZOP)

The initial process in any level of safety analysis is identifying the potential unintended behaviour that can occur in the system. The Hazard and Operability Study (HAZOP) process aims to help identify and analyse the hazards and operational concerns of a system. [10, p. 365]

In a safety-critical industry, HAZOP is identified as a structured process and technique where a multi-disciplinary team performs a systematic study of a system and its functions applying hazard guide words to discover deviations from the design intent and whether the consequences of these deviations can result in a hazard. [10]

This chapter uses the author's industry experience and the process defined by Ericson [10].

For consistency within this chapter, the term System will be used to refer

to the industry's use of both System and Feature to reflect current industry practices. A distinction between the terms will be made in Section 5.1.

## 2.1 Preliminary HAZOP Tasks

Before the team can begin executing the HAZOP process, preliminary work has to be performed to ensure that the inputs and pre-requisites to the HAZOP process have been met. This applies to any level of either functional or functional safety analysis. [10, pp. 368-370]

The LKA feature is a driver assistance system, which assists the driver actively in keeping the vehicle in lane by providing a steering torque overlay if an unintended lane departure is detected. If this steering overlay is not sufficient, then the LKA feature is intended to warn the driver to get the driver's attention back to vehicle control.

The LKA feature is meant to combine existing subsystems to generate a new feature. The lane-detection capability of the Front Camera Module (FCM) is used as well as the Electric Power Steering (EPS) capability of providing a torque overlay by an external request.

The LKA feature initiates different actions to keep the vehicle in lane and inform or warn the driver in different operational scenarios.

The LKA feature is intended as a fail-safe system.

The driver will be able to override every LKA initiated torque intervention by a counter steering maneuver.

The non-safety relevant LKA main function named "Control algorithm LKA" is allocated to the FCM.

All the safety relevant functions are allocated to the EPS and necessary peripheral sensing ECUs.

The LKA system considers that the driver always has hands on the wheel.

In order to avoid uncontrollability of the car, it is intended to use different dynamic and static vehicle driving states to identify critical driving situations. It is intended that during these critical situations no LKA intervention will be performed.

Figure 2.1: An Example of a Functional Description of the LKA System

Taking the Lane Keeping Assist (LKA) system as an example, Figure 2.1 contains the functional description of the LKA system defined by a multi-

disciplinary team and Figure 2.2 which shows the system concept and functional architecture prior to starting HAZOP.

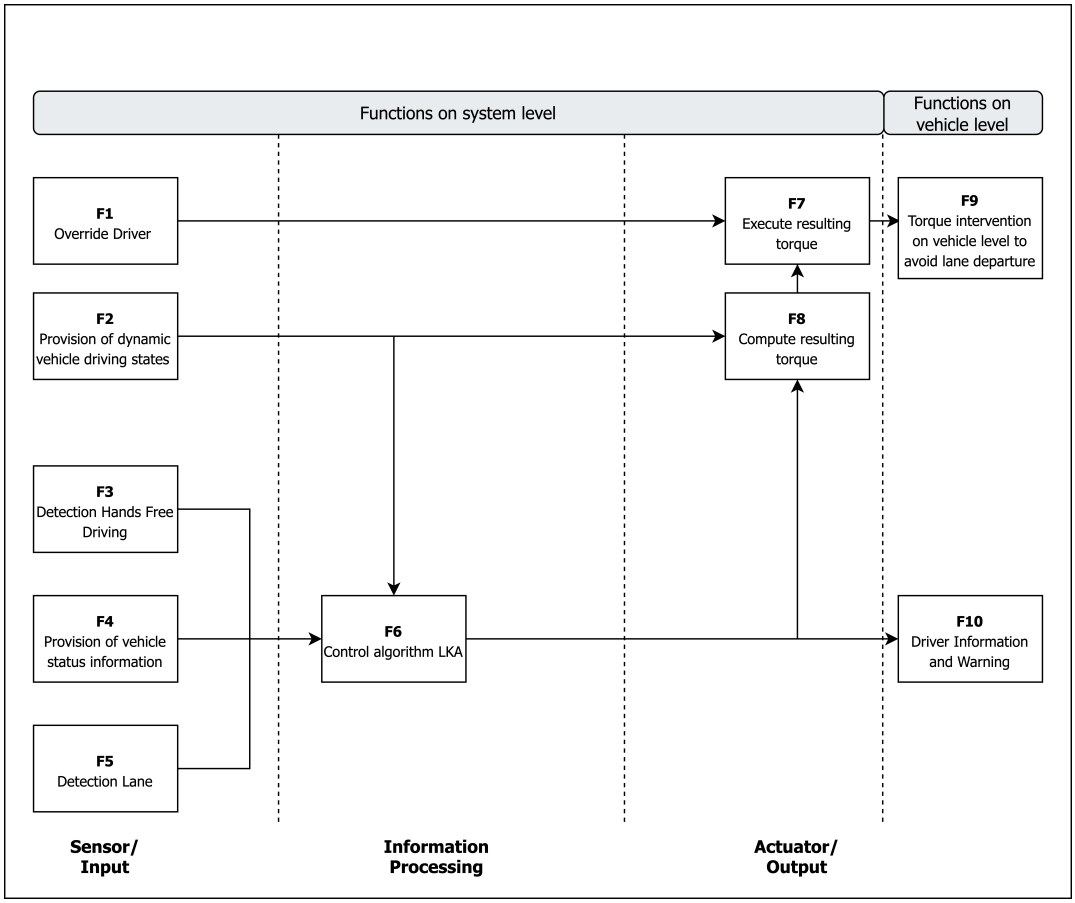



Figure 2.2: An Example of a LKA System Concept and Functional Architecture

The following subsections discuss the preliminary tasks required to be performed prior to executing HAZOP.

### 2.1.1 Defining the System Concept

To define and describe the system concept for HAZOP analysis, the team first defines the scope and boundaries of the system. The team defines the

system inputs and outputs (I/O), system functions, and the concept block diagram. Then, the team defines the potential applications of the system concept including vehicle platforms. Lastly, the team states and examines the assumptions about the vehicle features and functions.

### 2.1.2 Defining the Intended System Functions

The next step prior to performing HAZOP is to define all the intended high-level system functions in natural language.

The team provides a description of what the system should do and analyzes what parameter/s of the vehicle properties, kinematics, dynamics, etc. do the system outputs affect.

The team then decides how the system performs its functional goals and asks the following questions: Does the system use its own or does it rely on other systems' functions, sensors, and actuators? How much control does it have over other systems' resources?

The functional architecture allocation is shown in Figure 2.3. This figure contains the resulting allocation of the functional architecture in Figure 2.2 on the electro-mechanical control components and control computers (controllers).

### 2.1.3 Understand All Intended System Operating Conditions

The third step is to develop an understanding of all intended system operating conditions.

As defined in ISO-26262, a hazardous event is a combination of a hazard and an operational situation [14].

The hazards identified at a functional level are extended to meaningful

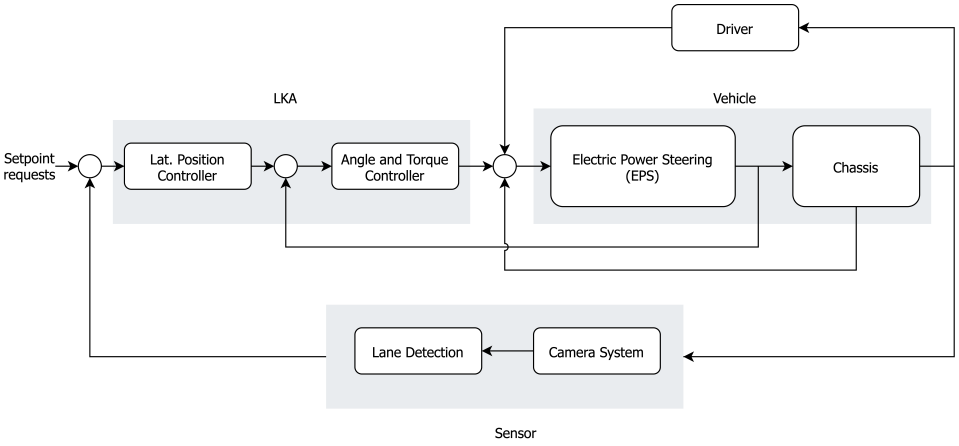



Figure 2.3: An Example of a LKA Functional Architecture Allocation

hazardous events used at the feature and vehicle level analysis by combining the hazards with operational situations.

The multi-disciplinary team takes on a system level approach to understanding how the system works in the vehicle during all intended operating conditions. This involves considering potential interactions with humans and other systems.

### 2.1.4 Understanding Intended System Functions and Behaviours During All Intended Operating Conditions

The last step in the process is to develop a clear understanding of the intended system functions and behaviours during all intended operating conditions.

The team considers the type, path, and interactions of energies in the system. The team also considers the different road and environment conditions and driving scenarios as shown in Figure 2.4.

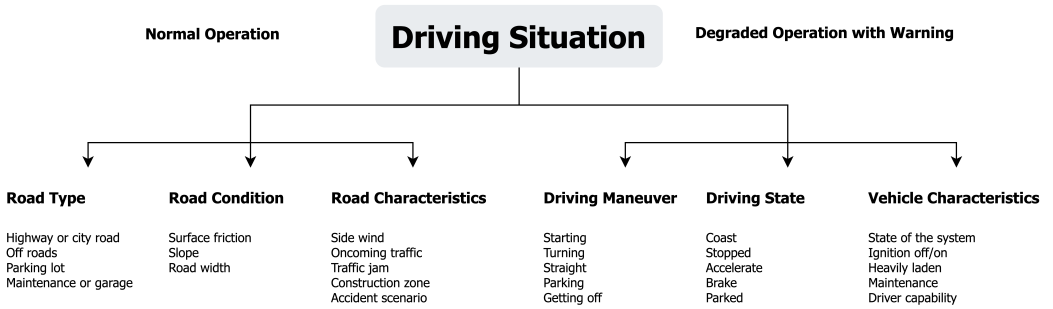



Figure 2.4: An Example of the Consideration of External Conditions and Driving Scenarios in the Automotive Industry

## 2.2 Performing HAZOP

After the pre-requisites have been fulfilled, only then can the team proceed with executing the HAZOP process.

### 2.2.1 Hazard Guide Words

The first step in executing HAZOP is to define hazard guide words [10]. To discover deviations from the design intent and whether the consequences of these deviations can result in a hazard or hazardous event, hazard guide words are applied to the statement of system functions.

To set an example, the following set of elementary/basic guide words:

- No Activation: System function not provided when needed.
- Autonomous Activation: System function provided when not needed.
- Excessive: System function provided more than needed or requested.
- Inadequate: System function provided less than needed or requested.

- Different Direction: System function provided in the opposite direction than needed or requested.
- Locked/Frozen: System function provided is locked, frozen, or fails to release.

This set of basic guide words are added to depending on the application and the level of functional and functional safety analysis.

Taking the real-time LKA system as an example, the addition of the following vehicle-level and temporal guide words was considered:

- Incorrect Activation: System function, when activated, leads to erratic vehicle behaviour
- Too Early: System function is provided too early.
- Too Late: System function is provided too late.
- Incorrect Activation/Deactivation: System function is in false mode.

### 2.2.2 HAZOP Execution

The outcome of the execution of HAZOP is presented in tabular format as shown in Figure 4.3 in Section 4.6.

The first row and first column contain the hazard guide words and system functions, respectively.

The set of hazard guide words are applied to the system functions to determine the foreseeable system reactions that are considered malfunctions. Malfunctions are considered to be reactions that either violate the intended system functions or cause safety concerns.

This exercise is performed implicitly with the understanding of the pre-requisites to HAZOP (system behaviour and operating conditions with respect to energies in the system) and is based on the team's own heuristics and expert opinions.

Each system level reaction is listed in $\text{cell}_{\text{function x hazard guide word}}$, where the convention denotes $\text{cell}_{\text{row x column}}$.

# Chapter 3

# Related Work

This chapter discusses the history, structure, and workflow of the generic traditional HAZOP process. Then, this chapter discusses instances of HAZOP as used in scientific research in chronological order. Execution of this study resulted in limited information as the HAZOP process is generally used in industry studies that are propriety in nature.

Traditionally, HAZOP is recognized and utilized as an effective systematic examination analysis method as part of the risk assessment process to identify hazards and hazardous events inherent and related to a system [8, 9, 10].

The traditional HAZOP requires that the analysis be performed by a large multi-disciplinary team of experts that have extensive knowledge of the design and operating scenarios of the system [10, 11].

In general, the HAZOP process is composed of these major steps:

- Understanding the system,
- Developing system models, and
- Hazard identification [8, 9, 10].

The first step in performing a general HAZOP analysis is to develop and convey a good understanding of the system. This will then be used to describe and convey the design intent of the entire system to the entire HAZOP team. Developing a good understanding of the system relies on the identification of the relevant operating scenarios and required behaviour of the interacting elements considered in the design [12].

The next step is to then develop system models based on the understanding of the system. Accurate and documented system models provide the basis for analysis required for HAZOP. The system models identify the elements of the system under consideration, their required functions and expected behaviours, and their interactions [8, 9, 13]. The system models serve as basis in conducting hazard identification.

Hazard identification is largely based on the following: the system models and the HAZOP team's expertise. Each system's element required functions and expected behaviours are analysed for malfunctions. Malfunctions are obtained by applying hazard guide words to the functions and behaviours and identifying the resulting deviations [10, p. 372]. From these malfunctions, resulting hazards and hazardous events are identified based on the interactions of the elements and operating scenarios.

According to Khan, *"The group of experts conducting HAZOP is helped along by 'guide words' which enables them to cover all possible malfunctions in the plant in a systematic way"* [11].

To summarize, traditional HAZOP is a largely ad hoc process that uses expert analysis in determining deviations from the required functions and behaviour to the resulting hazards and hazardous events.

Khan and Ericson summarized that performing traditional HAZOP is resource

intensive due to the requirement of a large expert multi-disciplinary team and the length of time required [11], [10, p. 370]. In addition to the required manpower to execute, HAZOP also yields results with high deviation, which require additional manpower to analyse.

Due to the amount of effort required to perform HAZOP, there have been multiple attempts to automate the HAZOP process. Parmar and Less have proposed an approach to systematically identify hazards by decomposing a system into its constituent units in their study of a closed water separator system [12]. These units and their relationships are then represented by block diagrams that model the plant. Each unit in the model is analysed for possible systematic faults initiated from deviations using traditional HAZOP. Each unit's initiated fault is then analysed against the plant model for transmission, propagation, and termination. Using rules and model analysis, Parmar and Less found that HAZOP requires much more than fault propagation. Though it plays an essential role in enhancing HAZOP, performing HAZOP using fault propagation requires a more comprehensive expert system.

Building on Parmar and Less' work, Khan proposed a knowledge-based expert system framework to conduct HAZOP [11]. Khan's proposal of an knowledge-based expert system software tool that identifies both general and process-specific causes and consequences of a system's constituent unit. Khan's proposed approach is to use a universal knowledge-base containing both general and process-specific knowledge. The general knowledge bank can therefore be used to perform HAZOP on any system whilst creating a different knowledge bank for the process-specific ones. Khan's approach starts with identifying the process variable deviations. Next, the deviation is applied to both upstream and downstream constituent units to find the general and process-specific

causes of the deviation. Then, analysis on the propagation of the deviation is performed to find the cause-consequence relationships for the system under study.

HAZOP has also been applied to the design of cyber security experiments [13]. Mansoori *et al* adapted a HAZOP methodology that focused on system and experimental design analysis application as well as risk and hazard mitigation. Mansoori et al identified points in the design where deviations are studied. By applying a set of guide words to describe the deviations to the component of the process, they identify the departure from the design intent. Taking a similar approach to Khan, Mansoori *et al* also identified the potential causes and consequences of the deviation. The potential hazard is expressed by the likelihood and severity if the deviation occurs. The required mitigating actions to remove the cause or eliminate the consequence are also identified. Mansoori *et al* have applied their HAZOP methodology in the study of malicious websites' IP tracking behaviour which resulted in the enhancement of their client's design.

A recent study performed on the Chinese Train Control System3 (CTCS-3) by Li *et al* in 2015 looked into how to guarantee the operational safety of the Chinese high-speed railway system using HAZOP [9]. The CTCS-3 onboard system is responsible for receiving the data and command information from the trackside then calculates the speed profile and safe operation of the moving train. Li *et al* developed four models within the study to describe the system design intent and operational situation. The models were also used to provide a basis for the hazard identification process. The corresponding models are as follows: reference model, functional hierarchical model, state diagram, and sequence diagram. After the completion of all these system

models, the hazards of the CTCS-3 onboard system can be identified during the examination session. Any deviations between the onboard system design intent and the operational situations are identified by experts on the basis of the systems model. The causes of deviations and consequences are also determined to help mitigate the risk and propose safety actions and measures. The introduction of this HAZOP study in the CTCS-3 onboard system has successfully identified high-accuracy hazards with respect to the increasing operational speed and expanding railway of the Chinese high-speed railway system.

# Chapter 4

# Evaluation of a typical Industry HAZOP Execution

This chapter evaluates the findings in the study of a typical industry HAZOP process.

While evaluating the sample process and its artefacts in Chapter 2, the following issues have been found:

- Use of the terms *Feature* and *System* interchangeably
- Fixed depth of System, Subsystem, and Component entities
- Inconsistent and unclear design basis
- Lack of traceability: Ad hoc, heuristic, and expert-based approach
- No central storage of information

## 4.1 Use of Feature and System Interchangeably

While evaluating the sample process and its artefacts, the use of the terms *Feature* and *System* both appeared to classify the LKA entity.

The use of the term Feature is driven by marketing jargon to refer to a new function that is considered outside the set of standard functionalities of a basic car.

On the other hand, the term System is used by the engineering team to define a set of vehicular entities that work together to achieve vehicle-level functional goals.

Review of the sample LKA Safety Case revealed that documentation teams use the terms System and Feature interchangeably to perform PHA-level HAZOP - immediate application of hazard guide words to system level functions to determine immediate feature-level reactions which are then used to determine system level hazards in the HAZOP spreadsheet in Chapter 4.6.

This creates confusion and ambiguity as the process was performed on what it seems to be different levels of abstraction.

Furthermore, the functional description does not provide distinctions between the two terms and uses them interchangeably as shown in Figure 2.1.

## 4.2 Fixed Depth of System, Subsystem, and Component entities

To continue the discussion in Section 4.1, Systems are broken down into lower level entities referred to as subsystems which have their own lower level functional goals. The lowest level entities are referred to as components.

The lower level functional goals are a decomposition of higher level functional goals and are allocated to lower level entities. Lower level functional goals help achieve higher level functional goals.

To summarize, the industry practices the following fixed-depth compositional hierarchy of entities: System, Subsystem, and Component. A System is composed of Subsystems which are then composed of Components.

This fixed abstraction depth takes away from developing an incremental safety case process that reduces the effort through impact scope analysis and assessment. Given a requirement/design change, the analysis will have to be performed at the fixed level and all atomic elements will have to undergo impact assessment via producer-consumer relationship.

If a dynamic abstraction depth achieved by functional decomposition is adhered to, the scope can be dynamic and the producer-consumer relationship can be done not necessarily at the atomic level, but at the higher functionally grouped level dictated by the decomposition. This is proposed in Chapter 8.

## 4.3 Inconsistencies and Lack of Clarity in the Design Basis

Further review of the sample documentation revealed the variations in the design details used as inputs to the design and safety design and analysis processes.

To have a better understanding of how the industry performs HAZOP, different stages of functional design and safety design have been evaluated.

The LKA logical view shown in Figure 4.1 and the system concept interface

shown in Figure 4.2 show deficiencies and conflicts with the functional architecture allocation in Figure 2.3. Though they are at different levels of abstraction, the signals to, from, and within the LKA entity and its components shall be consistent.

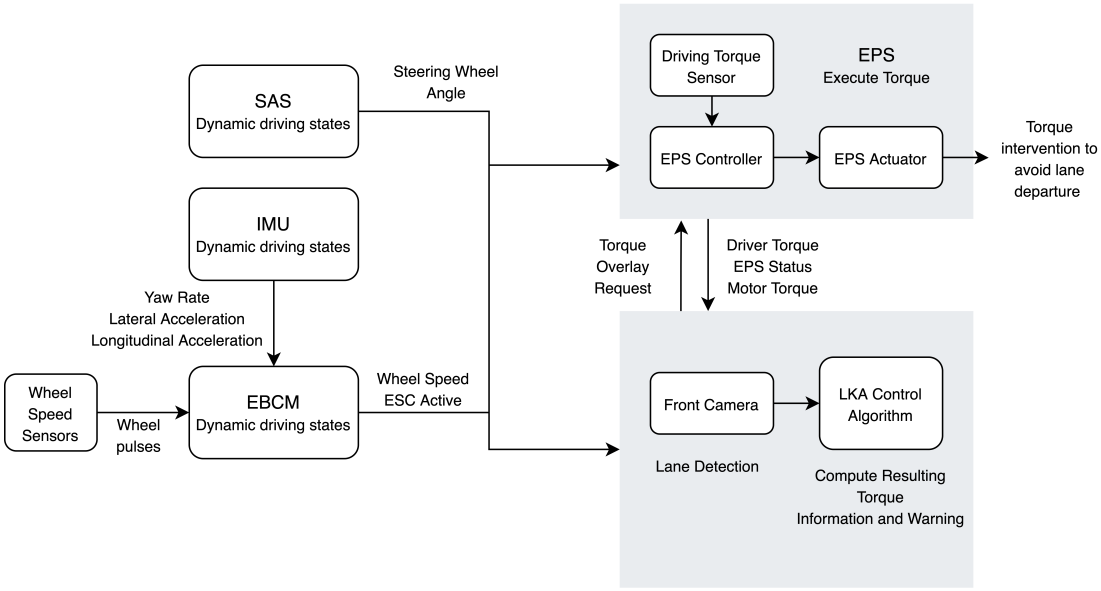



Figure 4.1: An Example of the LKA Design's Logical View

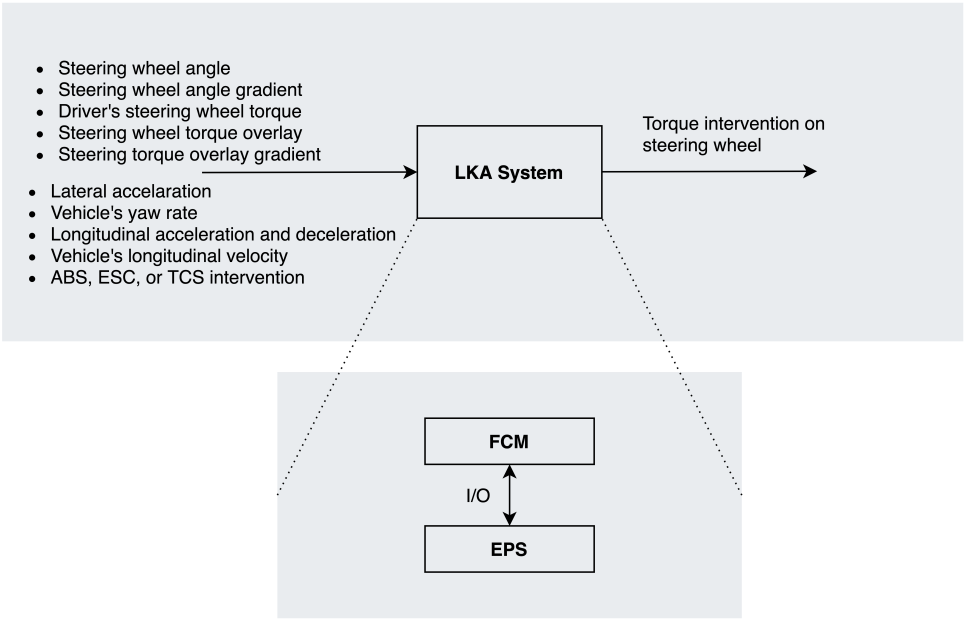



Figure 4.2: An Example of a System Concept Interface

Furthermore, the boundaries of the LKA entity have not been formally defined. Formal distinctions have not been made between shared and LKA-only resources, components, subsystems, and functionalities. Having no clear definitions and distinctions on scope and role of these items exponentially increase uncertainty and complexity in developing delta and incremental safety cases; Verification and validation activities are performed ad hoc and the change impact is considered to be at the atomic level.

## 4.4 Lack of Traceability: Ad Hoc, Heuristic, and Expert-Based Approach

Review of the HAZOP process in Chapter 2 shows that the HAZOP process heavily relies on the team members' heuristics, experiences, and expertise as

basis to determine the malfunctions and reactions of system functions to the hazard guide words.

This creates a problem: How is correctness evaluated? How can bias-free assurance be made?

## 4.5 No Central Storage of Information

Evaluating the artefacts reveals that multiple design bases exist which creates room for inconsistencies.

Assuming that the individual authors of each document are different, interpretations of the design details can vary greatly.

Maintenance, changes, and further updates to design create a waterfall effect on changes: Ideally, a change in one document shall require a change to others to keep the information consistent.

Due to constraints on time and resources, only a subset of the documents containing the information is updated to reflect the change/s.

## 4.6 Example HAZOP Execution Worksheet

| System Function vs. HAZOP Guidewords | Loss of Activation/ Sensing | Incorrect Activation/ Sensing (More than requested) | Incorrect Activation/Sensing (Less than requested) | Incorrect Activation/ Sensing (Activation in opposite direction) | Unintended Activation (when none was requested) | Locked Activation/ Sensing (Frozen function) | Incorrect Activation/ Sensing (Activation leads to erratic vehicle behaviour) | Incorrect Activation (too early) | Incorrect Activation (too late) | Incorrect Activation (in false mode) |
|---|---|---|---|---|---|---|---|---|---|---|
| Feature Reaction to HAZOP Guidewords | | | | | | | | | | |
| LKA intervention to maintain vehicle line | Loss of LKA feature | (1) Excessive wheel displacement; overcorrects causes opposite lane incursion with system relinquishing control afterward<br>(2) Excessive wheel rate; lateral jerk | Not enough correction (with either insufficient displacement or rate) Effective loss of LKA feature | Aggravates lane wandering. Creates lane incursion out of possible lane incursion with system relinquishing control afterward | Incursion into neighbouring lane, following or oncoming traffic with system relinquishing control afterward | Incursion into neighbouring lane, following or oncoming traffic, with locked steering control | Aggravates lane wandering. Creates lane incursion out of possible lane incursion with system relinquishing control afterward | (1) Excessive wheel displacement; overcorrects; causes opposite lane incursion with system relinquishing control afterward | Not enough correction (with either insufficient displacement or rate). Effective loss of LKA feature | (1) Unintended or unexpected LKA operation<br>(2) Unexpected or unintended loss of LKA operation |
| LKA lane detection | Loss of LKA feature | Unintended or unexpected LKA operation | Not enough correction (with either insufficient displacement or rate). Effective loss of LKA feature | N/A | Unintended or unexpected LKA operation | Unintended or unexpected LKA operation | N/A | N/A | N/A | Unintended or unexpected LKA operation |
| LKA vehicle path estimation | Loss of LKA feature | Unintended or unexpected LKA operation | Not enough correction (with either insufficient displacement or rate). Effective loss of LKA feature | N/A | Unintended or unexpected LKA operation | Unintended or unexpected LKA operation | N/A | N/A | N/A | Unintended or unexpected LKA operation |
| LKA disable | Unintended or unexpected LKA operation | N/A (Binary control/operation) | N/A (Binary control/ operation) | Unintended or unexpected LKA operation | Unintended or unexpected LKA operation | Unintended or unexpected LKA operation | Unintended or unexpected LKA operation | Effectively loss of LKA feature | Unintended or unexpected LKA operation | Unintended or unexpected LKA operation |
| LKA enable | Unintended or unexpected LKA operation | N/A (Binary control/ operation) | N/A (Binary control/ operation) | Unintended or unexpected LKA operation | Unintended or unexpected LKA operation | Unintended or unexpected LKA operation | Unintended or unexpected LKA operation | Unintended or unexpected LKA operation | Effectively loss of LKA feature | Effectively loss of LKA feature |
| LKA override | Inability to override lateral displacement causes lane incursion | If system needs more force to override than by design, then possible overcorrection into unintended lane | If system needs less force to override than by design, then effectively loss of LKA feature | If system needs opposite force to override than by design, then the driver effectively can't override | Effectively loss of LKA feature | Effectively loss of LKA feature | Effectively can't override | Effectively loss of LKA feature | Effectively can't override | N/A |
| LKA HMI | Effectively loss of LKA HMI feature | Unintended or unexpected LKA HMI Operation | Effectively loss of LKA HMI feature | Unintended or unexpected LKA HMI operation | Unintended or unexpected LKA HMI operation | Unintended or unexpected LKA HMI operation | Unintended or unexpected LKA HMI operation | Unintended or unexpected LKA HMI operation | Effectively loss of LKA HMI feature | Unintended or unexpected LKA HMI operation |

Figure 4.3: An HAZOP Execution Worksheet – Output of HAZOP Process

# Chapter 5

# HAZOP$^+$ Preliminaries

Prior to introducing the HAZOP$^+$ process, a few concepts should be discussed.

## 5.1 Feature vs. System

As evidenced in Section 4.1, the need to distinguish between Feature and System has to be addressed.

The definitions have been formulated based on current industry practices.

A Feature is a vehicle-level abstraction to classify a distinct vehicle-level functional entity that satisfies interrelated vehicle-level functional goals.

A System is a vehicle-level abstraction and classification concept to decompose and allocate, through exclusive ownership, Feature goals to functional physical/ hardware and abstract entities.

That is, a Feature has its own System of exclusive Resources (processors, sensors, actuators, etc.) that have dedicated functionality that works to address the Feature's goals.

Yet, a Feature may borrow or make use of other System's Functions and

Resources to accomplish its goals.

On the other hand, given a Feature A, it may lend its own System's resources to another Feature B to help achieve the Feature B's goals.

Figure 5.1 shows the relationship between System, Feature, and Resources.

The boundaries of Feature A (denoted by dashed lines) is a collection of its own functions FnA2 and FnA3 supplemented by FnB4 and FnC4 which are borrowed from System B and System C, respectively.

For Feature A to achieve its functional goals allocated to FnB4 and FnC4, it also also borrows RB2 and RC1 and from System B and System C, respectively.

The boundaries of System A are defined through the functions FnA2 and FnA3 and the functions' allocation to resources RA2 and RA4.

## 5.2 The 4-Variable Model

To provide context to building functional requirements and to have a proper system model analysis, the 4-variable model is introduced.

Figure 5.2 illustrates a Control System containing software as viewed from a general control theory approach.

The components of the Plant and Environment that are monitored or observed are referred to as *monitored variables* (MV) and those components that are controlled or affected are referred to as *controlled variables* (CV).

The Control System's Input and Output Boundary Functions, such as the functions of sensors and actuators, to interact with the MVs and CVs.

The Control System monitors MVs as system inputs with the use of hardware sensors, performs its functions, and controls the CVs as system outputs with the use of hardware actuators. The Control System's logical functions are

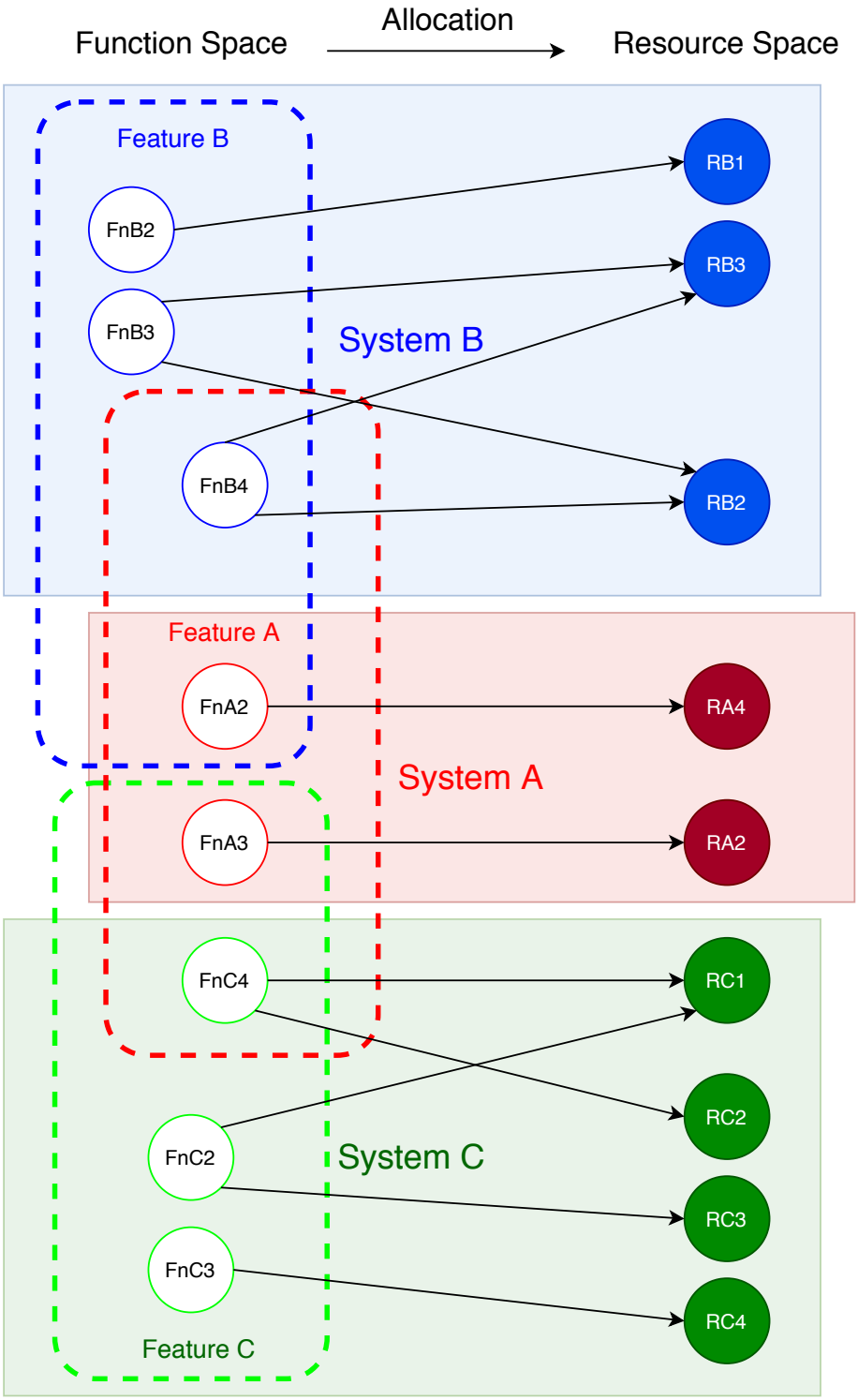



Figure 5.1: System, Feature, and Resources

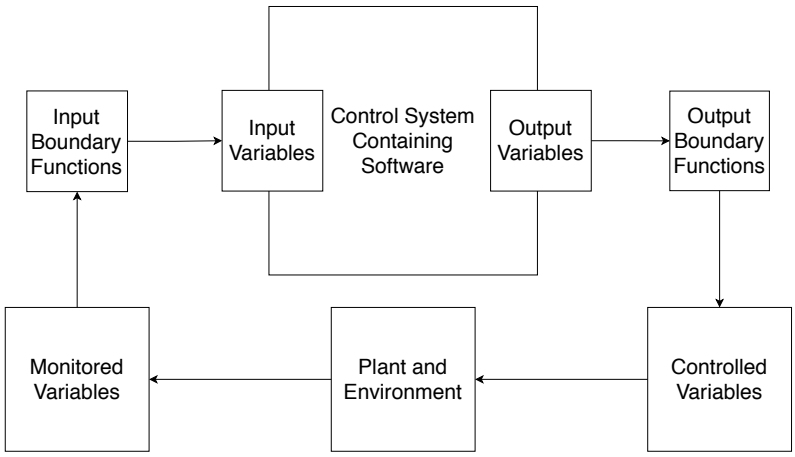



Figure 5.2: The 4-Variable Model

governed by the embedded software.

## 5.3 Car-Driver-Environment Considerations

Ideal modelling of systems and interactions involves the consideration of all interacting elements (*universe*) down to the atomic level.

In practice, it is not feasible to perform such a complex and detailed task due to the number of elements, attributes, and minute interactions to consider.

The scope of analysis shall be kept at a reasonable level. The scope shall also include all pertinent elements in the system's universe. As such, the following are considered at the highest level of abstraction:

- Car
- Driver
- Environment

The Car-Driver-Environment model presented in Figure 5.3 shows the general interactions between the Car, Driver, and Environment as single-line inputs and outputs.

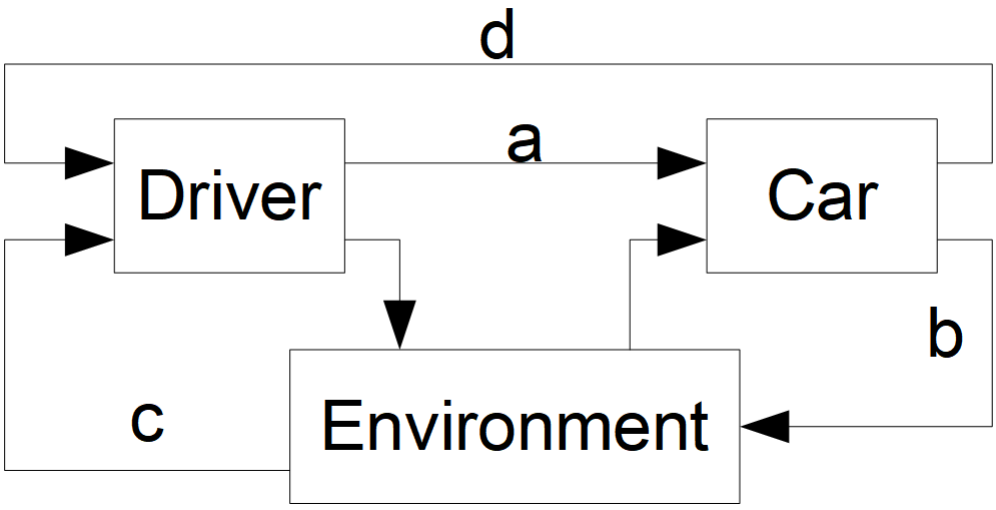



Figure 5.3: The Car-Driver-Environment Model

## 5.4 Function Traversal

In Section 5.2, the concept of a system's boundary functions was introduced.

*Boundary Functions* serve as interface points for either input or output signals to connect to a model's element as shown in Figure 5.2.

*Function Traversal* is an activity where each functional element in a model is sequentially traversed using the input and output boundary functions as reference and in the direction the arrows point to which indicate the signal paths.

A *Producer* sends a signal to be received by a *Consumer*.

In Figure 5.3, the Driver (Producer) sends signal $a$ to the Car (Consumer) as indicated by the direction of the arrow. The Driver treats signal $a$ as an

output, while the Car treats signal $a$ as an input.

Moving the reference of the analysis to Car and Environment, the Car acts as the Producer of signal $b$ to be received by the Environment as the Consumer.

## 5.5 The Concept of Functional Requirements

*Functional Requirements* are formal statements of required functions that the subject shall accomplish.

The requirements are formulated such that they place an imperative onus on the subject to provide the essential functional goals.

Statements of functional requirements shall be:

- Clear,
- Concise,
- Consistent,
- Testable,
- Observable (inputs and outputs can be observed),
- and Free of Ambiguity.

Each statement of a functional requirement shall have the following elements:

- Monitored variable/s and/or
- Controlled variable/s,
- Goal to be accomplished, and
- Subject of the requirement (i.e. the feature).

There are multiple ways to state functional requirements. Depending on the safety rating of the system, it may be required to use formal methods to state functional requirements. Formal methods include function tables and mathematical expressions.

For the purposes of this thesis, functional requirements are stated in natural language. An example is listed below:

*The ABC Feature (Subject) shall turn on (Goal) the electrical power (CV) when the key switch position (MV) is placed in start.*

# Chapter 6

# PHA-level HAZOP$^+$

This chapter discusses the proposed systematic approach HAZOP$^+$ at the level of Preliminary Hazard Analysis (PHA) and how it can be extended to other levels of abstraction.

The following is the summary of executing HAZOP$^+$:

- Define functional requirements (Section 6.1)
- Develop FAD (Section 6.2)
- Define applicable HAZOP Guide Words (Section 6.3)
- Perform FFA (Section 6.4.1)
- Perform BTA (Section 6.4.2)
- Perform BNT (Section 6.4.3)
- Determine Feature Reaction (Global Badness to Requirements Failure) (Section 6.5)

- Determine System-Level Hazards by extending the Feature Reactions/Malfunctions to the Vehicle-Feature-Driver-Environment interactions (Section 6.5)

## 6.1 Feature-level Functional Requirements Definition

The first step in the proposed approach is to formulate the top-level functional requirements for the subject.

As defined in Section 5.5, the functional requirements shall be formulated with consideration of the Car-Driver-Environment universe interactions.

PHA is performed in the initial stages of Feature design. Therefore, the analysis' top-level of abstraction shall also be at the Feature level.

The following subsections define the steps to define Feature-level Function Requirements.

### 6.1.1 Car-Driver-Environment-Feature Block Diagram

As Feature is the subject of this analysis, it shall be integrated into the model of the universe defined in Section 5.3. The resulting Figure 6.1 shows the reference point of Feature analysis with Feature embedded in the Car.

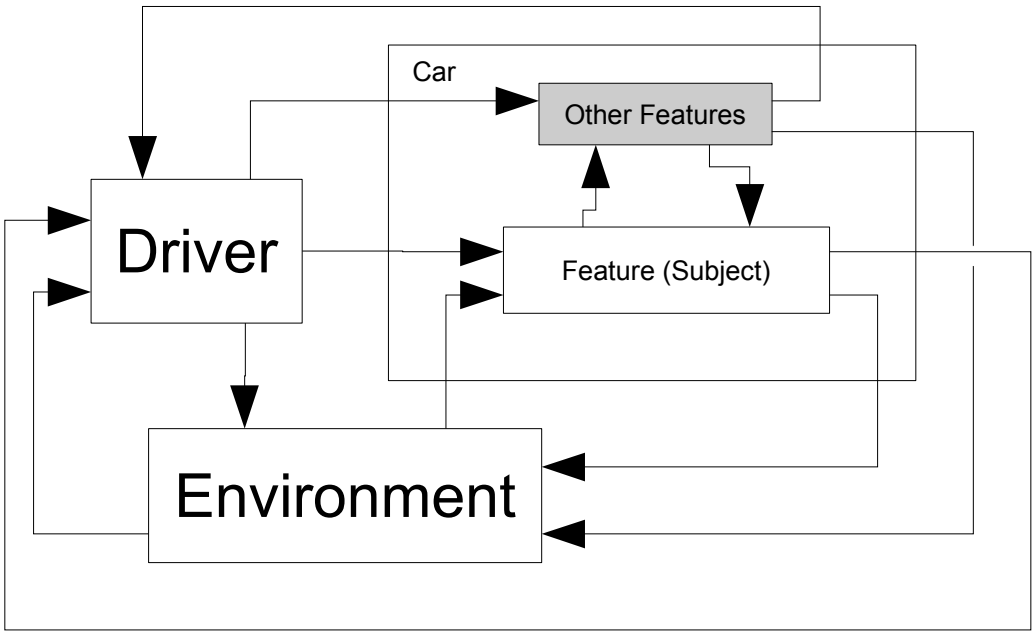



Figure 6.1: Initial Car-Driver-Environment-Feature Diagram

Figure 6.2 shows the single block diagram of Feature and its inputs and outputs. Feature interacts with the components of Car, Driver, Environment, and Other Features by receiving input signals and sending output signals. The signals are shown as single-line I/O for the purpose of cleanliness, but it can be further decomposed depending on the level of analysis.

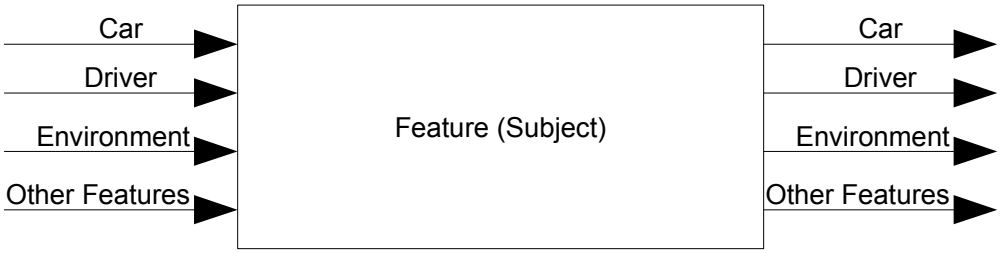



Figure 6.2: Car-Driver-Environment-Feature Diagram w/ Feature as Subject

### 6.1.2 Identifying Expected Functionalities

The multi-disciplinary design team formulates the expected functionalities of the Feature with inputs from the marketing team. The design team shall develop an abstract idea of which elements of the model are involved and how they interact using observable effects and flow of data and energies through inputs and outputs.

### 6.1.3 Identifying Monitored and Controlled Variables

After defining the Feature's expected functionalities in Section 6.1.2, the team shall formally identify the monitored and controlled variables (MVs and CVs) of the Feature. Section 5.2 and Figure 5.2 state the relationship that is defined by the Computer System (Feature), the MVs and CVs, and the Plant and Environment (Car-Driver-Environment).

### 6.1.4 Defining Feature Boundary

At this point, the team shall define the Feature boundaries starting with the boundary functions (sensors and actuators) as it relates to the Car-Driver-Environment-Feature model and the statement of MVs and CVs.

The team shall relate how the Feature observes MVs as inputs and affects CVs as outputs with the use of input and output boundary functions.

Figure 6.1 shall be updated with the specific boundary functions that the Feature uses resulting in Figure 6.3.

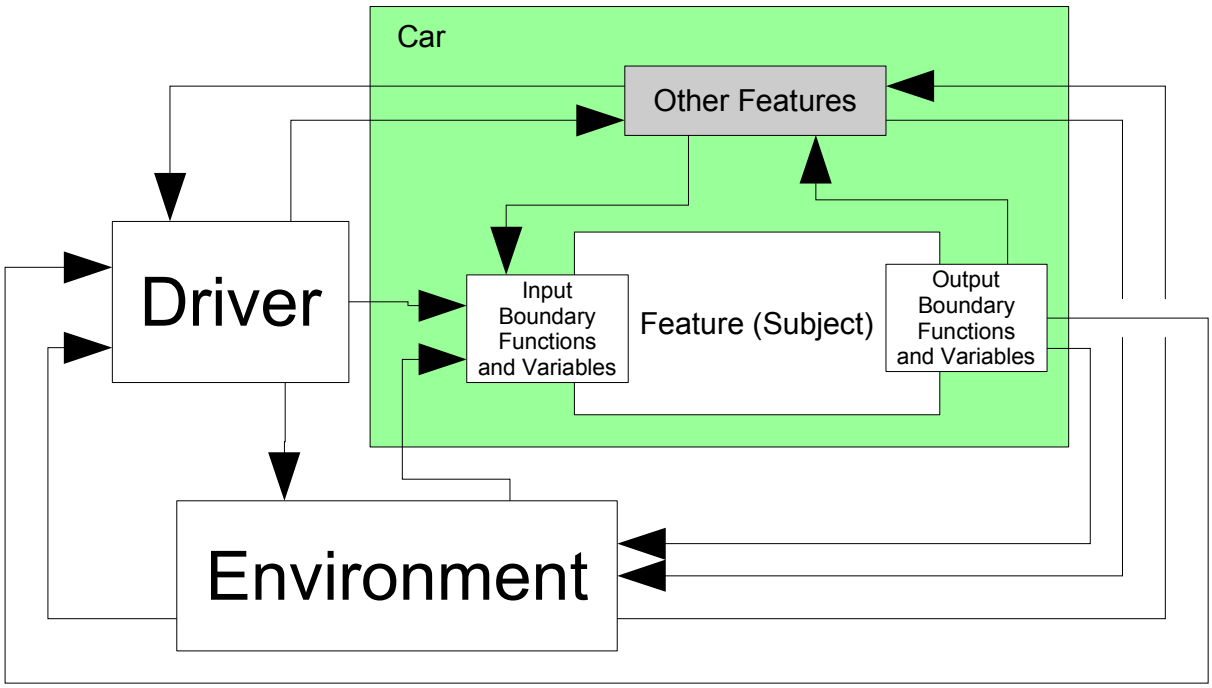



Figure 6.3: Car-Driver-Environment-Feature Diagram w/ Feature Boundary

### 6.1.5 Statement of Functional Requirements

After all the pre-requisites have been fulfilled, only then can functional requirements be formally defined. Section 5.5 defines the concept of functional requirements. Each required functionality shall be stated in one or more functional requirements.

## 6.2 Functional Architecture Diagram (FAD)

This section describes the process for creating the Functional Architecture Diagram (FAD). The FAD is used in the execution of HAZOP as basis for *"badness"* (malfunction/failure) creation and transfer, hazards traceability, and change impact analysis.

### 6.2.1 Functional Blocks from Functional Requirements

Section 6.1 describes the process of defining functional requirements at the Feature level. Using the statements of functional requirements, functional blocks are created to provide a visual representation of the Feature's functional elements. Each functional requirement shall be provided in one or more functional blocks.

From the sample discussion in Section 5.1 and Figure 5.1, the functional blocks are shown in Figure 6.4 for Feature A.


Figure 6.4: Functional Blocks of Feature A

### 6.2.2 Functional Network

The dependencies (producer and consumer relationships) of the functional blocks obtained in Section 6.2.1 shall be analysed and visually represented through a network of functions. Figure 6.5 shows a sample functional network illustrating the dependencies between the sample function blocks.

- An element outside of the figure (Consumer) depends on FnC4 (Producer).
- FnC4 (Consumer) depends on both FnA2 and FnA3 (Producers).
- FnA3 (Consumer) depends on both FnA2 and FnB4 (Producers).
- FnA2 (Consumer) depends on FnB4 (Producer).

- FnB4 (Consumer) depends on an element outside of the figure (Producer).

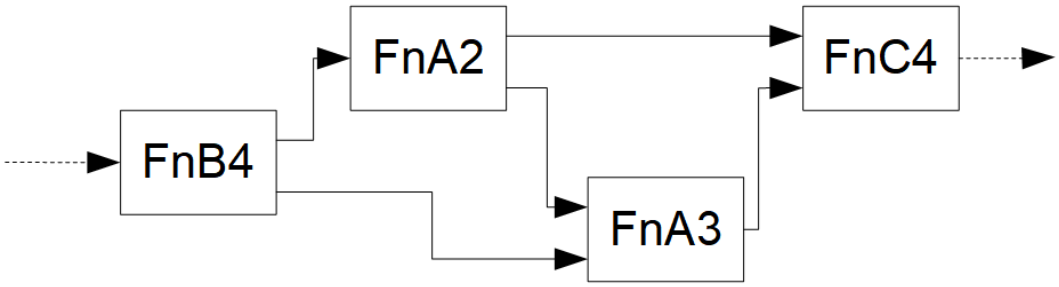



Figure 6.5: Functional Network

### 6.2.3 Feature Input and Output Boundary Functions w/ FAD

After creating the functional network in Section 6.2.2, the input and output boundary functions of the Feature are identified by the leftmost and rightmost function blocks, respectively.

Figure 6.6 is the resulting Functional Architecture Diagram. It shows the integration of the Functional Network in Figure 6.5 with the Car-Driver-Environment-Feature w/ Feature Boundary model in Figure 6.3. The input and output boundary functions of the Feature are FnB4 and FnC4, respectively.

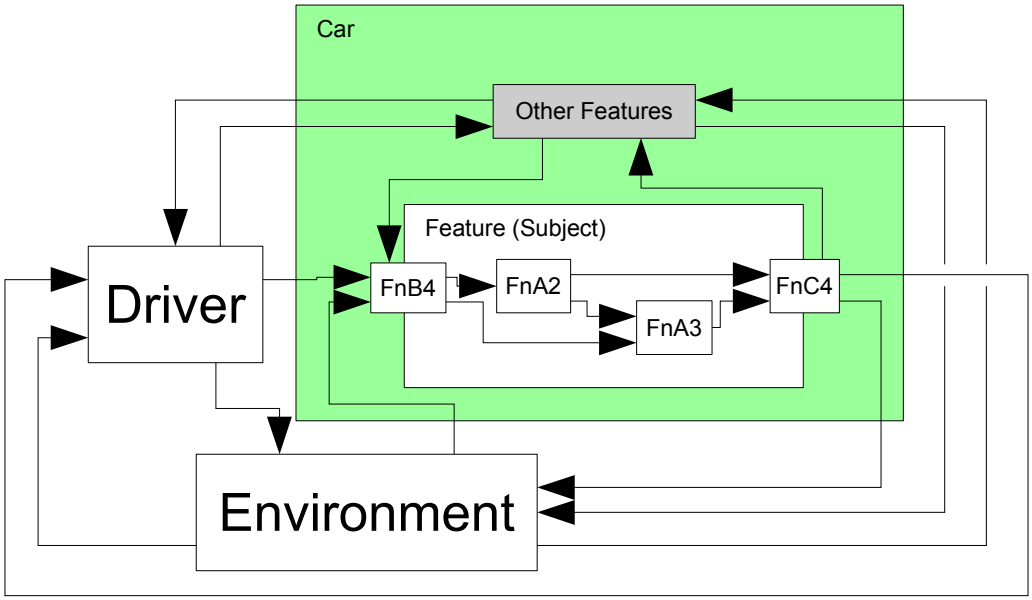



Figure 6.6: Functional Architecture Diagram of Feature A

## 6.3 HAZOP Guide Words

Section 2.2.1 provides an overview of how HAZOP Guide Words are used in industry practice. HAZOP Guide Words are used to discover deviations from the design intent and whether the consequences of these deviations can result in a hazard or hazardous event. The following subsections define the basis and use of guide words in HAZOP.

### 6.3.1 Basic Guide Words

For consistency, a set of basic guide words shall be created for common use across all Features related to a car.

The proposed systematic analysis will be performed simultaneously with traversing the Functional Architecture Diagram (Section 6.2). Therefore, the analysis will be done at the I/O signal level.

Similar to the approach in Section 2.2.1, the following are used to build the basic set of guide words that focus on signal level deviation/malfunction:

- No Activation: System function not provided when needed.
- Autonomous Activation: System function provided when not needed.
- Excessive: System function provided more than needed or requested.
- Inadequate: System function provided less than needed or requested.
- Different Direction: System function provided in the opposite direction than needed or requested.
- Locked/Frozen: System function provided is locked, frozen, or fails to release.

### 6.3.2 Additional Guide Words

Feature specificity, the level of abstraction, and type of application dictate feasible options for additional hazard guide words.

For example, real-time systems consider the addition of guide words that are of temporal type.

## 6.4 Function and Failure Analysis: Badness Creation and Transferring

This subsection discusses the consideration of the following two types of failures: inherent failure and transferred badness. This subsection also considers the

analysis approach for each of the types: Function Failure Analysis (FFA) and Badness Transfer Analysis (BTA).

The main goals of HAZOP are to consider what internal badnesses in the system can occur if a function in FAD fails, how those badnesses are transferred to the boundary functions, and which of system requirements will fail. Thus, the following shall be discussed:

- badness creation (FFA in Section 6.4.1),
- badness transfer by a single function (BTA in Section 6.4.2),
- composition of those local transfers into a global badness transfer over a network (BNT in Section 6.4.3), and
- transferring of badnesses onto requirement failures in Section 6.5.

### 6.4.1 Function Failure Analysis (FFA)

Function Failure Analysis (FFA) defines the set of all possible badnesses that could appear at all of a given function's outputs if the function fails.

Initially define a function $f1$ having the internal functional elements $g1$, $g2$, and $g3$, such that $f1$ is composed of $g1$, $g2$, and $g3$. Function $f$ also has the outputs $o1$, $o2$, and $o3$.

Given a scenario where failure occurs in $g1$ through incorrect design (i.e. mistake in programming or wrong assumption), the failure is observed and realised through the outputs of $f1$.

As such, the composition of $f1$ will not be dealt upon at this level of analysis (black-box approach). Instead, FFA will be performed on the output signals.

FFA is applying the set of hazard guide words onto a function's outputs to determine the badnesses related to the function.

Using the function $f1$ defined above, Table 6.1 shows the tabular representation of FFA performed for $f1$ and additional $f2$ and $f3$.

Table 6.1: Sample FFA Table

| Function-Signal \ Guide Word | | gw1 | gw2 | gw3 |
|---|---|---|---|---|
| f1 | o1 | gw1 $\otimes$ o1 | gw2 $\otimes$ o1 | gw3 $\otimes$ o1 |
| | o2 | gw1 $\otimes$ o2 | gw2 $\otimes$ o2 | gw3 $\otimes$ o2 |
| | o3 | gw1 $\otimes$ o3 | gw2 $\otimes$ o3 | gw3 $\otimes$ o3 |
| f2 | o4 | gw1 $\otimes$ o4 | gw2 $\otimes$ o4 | gw3 $\otimes$ o4 |
| f3 | o5 | gw1 $\otimes$ o5 | gw2 $\otimes$ o5 | gw3 $\otimes$ o5 |
| | o6 | gw1 $\otimes$ o6 | gw2 $\otimes$ o6 | gw3 $\otimes$ o6 |

The $\otimes$ operator denotes the resulting failure or reaction by applying the guide word onto the output signal.

Each hazard guide word is applied to each signal and the resulting badness is the by-product of the cell elements $\text{cell}_{\text{function-signal x hazard guide word}}$, where the convention denotes $\text{cell}_{\text{row x column}}$.

An example of translation to a statement in natural language shall be:

$$\text{gw1} \otimes \text{o1} \rightarrow \text{Excessive (gw1) force output request (o1)}$$

### 6.4.2 Badness Transfer Analysis (BTA)

The next step is to determine the badness transfer of a single function. Badness Transfer Analysis (BTA) determines the effect of input badnesses on a function's

outputs (i.e. How will a function's outputs fail if its inputs are bad?). BTA identifies a function's transferred badnesses which are the effect of failures/badnesses of pre-requisite input functions and signals.

Each function reaction to an input badness will be described through the function's outputs. Table 6.2 shows BTA performed on input $i1$ of function $f1$.

Table 6.2: Sample BTA Table for Input $i1$ of Function $f1$

| | | f1: i1 | | |
|---|---|---|---|---|
| Signal \ Guide Word | | gw1 | gw2 | gw3 |
| f1 | o1 | i1,gw1 ⊗ o1 | i1,gw2 ⊗ o1 | i1,gw3 ⊗ o1 |
| | o2 | i1,gw1 ⊗ o2 | i1,gw2 ⊗ o2 | i1,gw3 ⊗ o2 |
| | o2 | i1,gw1 ⊗ o2 | i1,gw2 ⊗ o2 | i1,gw3 ⊗ o2 |

Similar to Table 6.1, the ⊗ operator denotes the resulting failure or reaction by applying the guide word onto the output signal.

Each hazard guide word driven by the input signal is applied to each output signal and the resulting badness is the by-product of the cell elements $\text{cell}_{\text{output signal x input,hazard guide word}}$, where the convention denotes $\text{cell}_{\text{row x column}}$.

An example of translation to a statement in natural language shall be:

i1,gw1 ⊗ o1 → Excessive (gw1) incline detection value (i1) yields excessive (badness transferred) force output request (o1)

### 6.4.3 FFA + BTA + FAD = Badness Network Transfer (BNT)

Badness Network Transfer (BNT) is the analysis process of combining FFA, BTA, and FAD to determine the causal relationships and network of badnesses of the Feature.

The results of FFA and BTA are extended to the traversal of the Feature FAD (Section 5.4). Functional dependencies (defined by I/O signal relationships in the network: Section 6.2.2) are analysed and provided context along with the FFA and BTA badnesses.

The analysis shall begin at the output boundary functions of the Feature. The badnesses (inherent function failures from FFA and transferred badnesses from BTA) at the output boundary functions are added to the set of Feature-level badnesses.

Next, the connections between the output boundary functions and its input functions, through I/O signals, are analysed by means of consumer-producer relationships (Section 6.2.2) such that apparent badnesses of the consumer functions are attributed to either an inherent consumer function badness or transferred badness from the pre-requisite producer function.

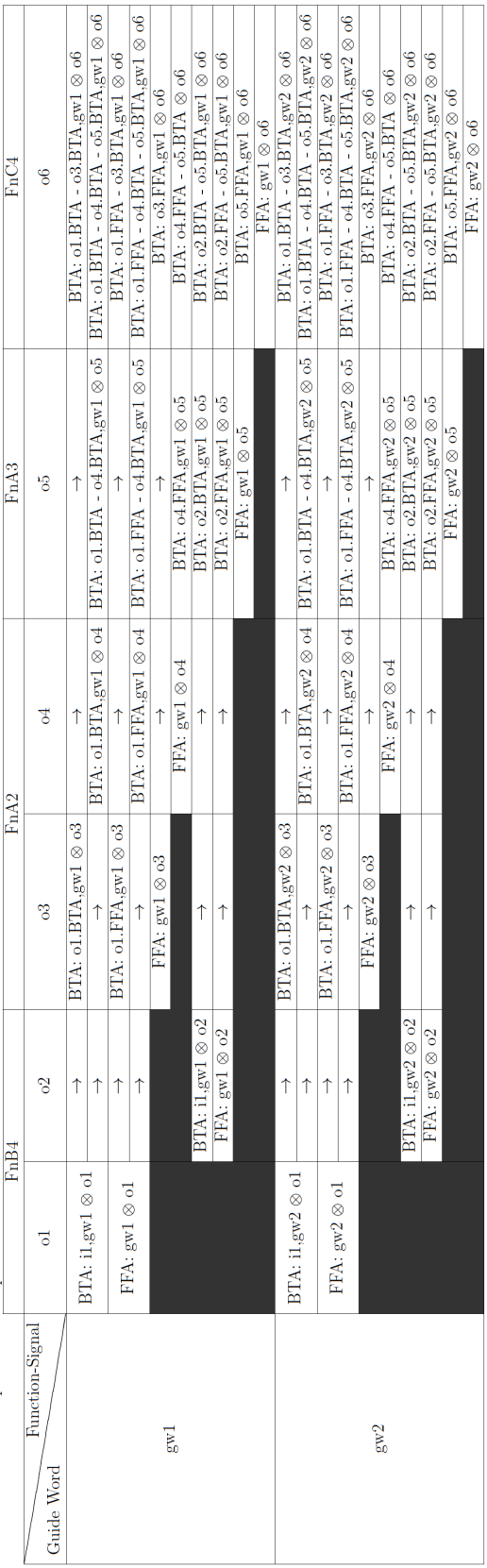
Analyses of consumer-producer relationships through traversals of the Feature FAD are performed until the input boundary functions are reached.

The FAD in Figure 6.6 is used as reference for a sample case: Given FnB4, FnA3, and FnC4 are inherently correct and that there exists an inherent badness in FnA2 (producer), the transferred badnesses of FnA3 and FnC4 (consumers) are attributed to the inherent badness/es originating from FnA2.

### 6.4.4 Tabular Representation of BNT Results

The results of BNT analysis can be documented in a variety of ways. For consistency, the tabular approach will be presented.

Figure 6.7 shows the results of executing a sample BNT analysis in tabular format using two hazard guide words. The analysis is performed on the FAD in Figure 6.8 which shows the I/O paths and function dependencies.



| Guide Word | Function-Signal | FnB4 | | FnA2 | | FnA3 | FnC4 |
|---|---|---|---|---|---|---|---|
| | | o1 | o2 | o3 | o4 | o5 | o6 |
| gw1 | BTA: i1,gw1 ⊗ o1 | | → | BTA: o1.BTA,gw1 ⊗ o3 | → | → | BTA: o1.BTA - o3.BTA,gw1 ⊗ o6 |
| | | | → | → | BTA: o1.BTA,gw1 ⊗ o4 | BTA: o1.BTA - o4.BTA,gw1 ⊗ o5 | BTA: o1.BTA - o4.BTA - o5.BTA,gw1 ⊗ o6 |
| | FFA: gw1 ⊗ o1 | | → | BTA: o1.FFA,gw1 ⊗ o3 | → | → | BTA: o1.FFA - o3.BTA,gw1 ⊗ o6 |
| | | | → | → | BTA: o1.FFA,gw1 ⊗ o4 | BTA: o1.FFA - o4.BTA,gw1 ⊗ o5 | BTA: o1.FFA - o4.BTA - o5.BTA,gw1 ⊗ o6 |
| | | | | FFA: gw1 ⊗ o3 | → | → | BTA: o3.FFA,gw1 ⊗ o6 |
| | | | BTA: i1,gw1 ⊗ o2 | → | FFA: gw1 ⊗ o4 | BTA: o4.FFA,gw1 ⊗ o5 | BTA: o4.FFA - o5.BTA ⊗ o6 |
| | | | FFA: gw1 ⊗ o2 | → | → | BTA: o2.BTA,gw1 ⊗ o5 | BTA: o2.BTA - o5.BTA,gw1 ⊗ o6 |
| | | | | → | → | BTA: o2.FFA,gw1 ⊗ o5 | BTA: o2.FFA - o5.BTA,gw1 ⊗ o6 |
| | | | | | | FFA: gw1 ⊗ o5 | BTA: o5.FFA,gw1 ⊗ o6 |
| | | | | | | | FFA: gw1 ⊗ o6 |
| gw2 | BTA: i1,gw2 ⊗ o1 | | → | BTA: o1.BTA,gw2 ⊗ o3 | → | → | BTA: o1.BTA - o3.BTA,gw2 ⊗ o6 |
| | | | → | → | BTA: o1.BTA,gw2 ⊗ o4 | BTA: o1.BTA - o4.BTA,gw2 ⊗ o5 | BTA: o1.BTA - o4.BTA - o5.BTA,gw2 ⊗ o6 |
| | FFA: gw2 ⊗ o1 | | → | BTA: o1.FFA,gw2 ⊗ o3 | → | → | BTA: o1.FFA - o3.BTA,gw2 ⊗ o6 |
| | | | → | → | BTA: o1.FFA,gw2 ⊗ o4 | BTA: o1.FFA - o4.BTA,gw2 ⊗ o5 | BTA: o1.FFA - o4.BTA - o5.BTA,gw2 ⊗ o6 |
| | | | | FFA: gw2 ⊗ o3 | → | → | BTA: o3.FFA,gw2 ⊗ o6 |
| | | | BTA: i1,gw2 ⊗ o2 | → | FFA: gw2 ⊗ o4 | BTA: o4.FFA,gw2 ⊗ o5 | BTA: o4.FFA - o5.BTA ⊗ o6 |
| | | | FFA: gw2 ⊗ o2 | → | → | BTA: o2.BTA,gw2 ⊗ o5 | BTA: o2.BTA - o5.BTA,gw2 ⊗ o6 |
| | | | | → | → | BTA: o2.FFA,gw2 ⊗ o5 | BTA: o2.FFA - o5.BTA,gw2 ⊗ o6 |
| | | | | | | FFA: gw2 ⊗ o5 | BTA: o5.FFA,gw2 ⊗ o6 |
| | | | | | | | FFA: gw2 ⊗ o6 |

Figure 6.7: Sample Table of BNT Results

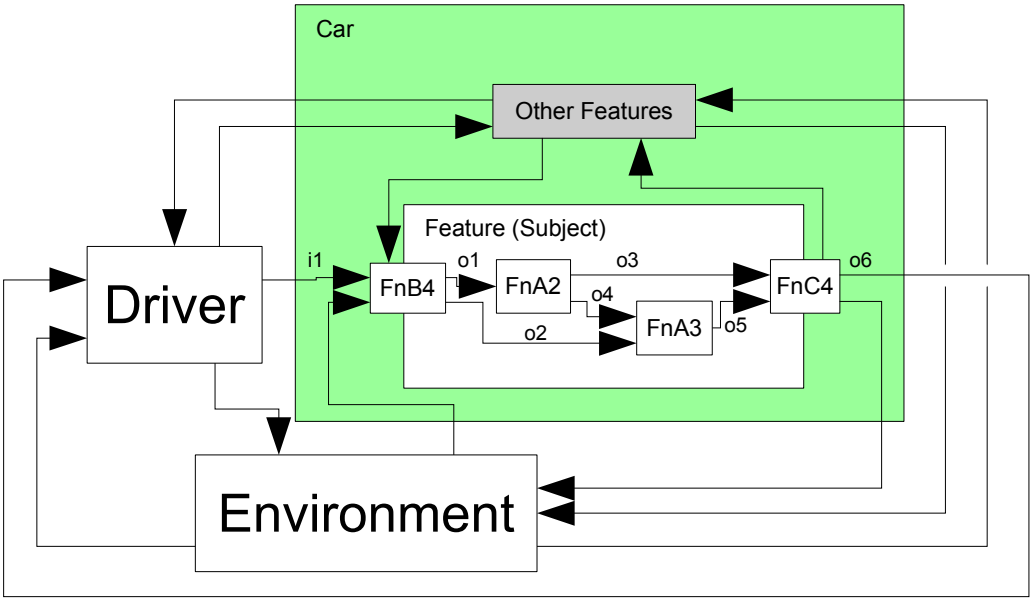



Figure 6.8: Functional Architecture Diagram of Feature A w/ I/O

To interpret the results, the table is traversed from left to right.

FnB4's o1 output is dependent on i1. Its BTA analysis against gw1 (BTA: i1,gw1 $\otimes$ o1) is recorded in $\text{cell}_{\text{gw1 x o1}}$.

Since o2 is independent from o1, a right arrow ($\rightarrow$) is placed to the right of BTA: i1,gw1 $\otimes$ o1 indicating a "skip."

The analysis is next performed on o3. BTA is performed on o3 against gw1 as the result of the transferred badness from o1. This is indicated by BTA: o1.BTA,gw1 $\otimes$ o3 in $\text{cell}_{\text{gw1 x o3}}$ that is in the same row as BTA: i1,gw1 $\otimes$ o1.

The analysis is then extended to o6 as indicated by BTA: o1.BTA - o3.BTA,gw1 $\otimes$ o6. The notation shows that the BTA analysis is performed against gw1 on the result of o1.BTA to o3.BTA.

Similarly, FFA is performed on o1 as indicated by FFA: gw1 $\otimes$ o1. This badness is transferred to o3 by BTA: o1.FFA,gw1 $\otimes$ o3. This is then extended to BTA: o1.FFA - o3.BTA,gw1 $\otimes$ o6.

## 6.5 Global Badness to Requirements Failure

Execution of the BNT process provides the malfunctions on an I/O level. The naming convention used for the I/O signals shall encourage meaningful semantics. Instead of ad hoc variable names such as o1 and o2, variable names shall provide context as to what they represent and their application.

Feature function requirements are defined against MVs and CVs (Section 5.5). In the FAD shown in Figure 6.8, the variables i1 and o6 are the MV and CV, respectively.

Substituting i1 and o6 with semantic variable names such as *m_force_detected* and *c_force_pid_out*, badnesses obtained in the BNT analysis now observe semantic traceability to the feature functional requirements.

Given a scenario where FnB4 produces a badness and transfers the failure all the way to FnC4, the requirements associated with the impacted functions are now considered *failed functional requirements*.

The failed requirements, though given piecemeal interpretation, are now semantically connected by the functional network described in its FAD (Section 6.2).

Failures of requirements are confirmed and described by the Feature's functionality and observability. Requirements at the boundary functions (input and output) are given primary notice in semantically connecting the failed requirements.

Boundary function requirements state the MVs and CVs in its scope and can be observed outside the Feature boundary. The MVs and CVs also provide direct interaction with the outside elements of the Car-Driver-Environment-Feature model.

# Chapter 7

# An Instance of PHA-level HAZOP$^+$ on LKA

This chapter shows an execution of PHA-level HAZOP$^+$ on the Lane Keeping Assist (LKA) Feature.

## 7.1 LKA Feature-level Functional Requirements Definition

Table 7.1 shows the Feature-level functional requirements that have been obtained from details contained in Chapter 2. These are deemed the basic conceptual design requirements and will require further requirement in the detailed design stages.

Table 7.1: Feature-level Functional Requirements of the LKA

| # | Requirement |
|---|---|
| FN-1a | When the LKA feature and car are on and the driver is crossing the lane marking unintentionally, the LKA shall provide torque intervention to maintain the vehicle in its driving lane by moving the vehicle laterally as intended. |
| FN-1b | The LKA feature shall not intervene if the driver is crossing the lane marking intentionally. |
| FN-1c | The LKA feature shall detect if the driver is intentionally or unintentionally crossing the lane marking. (Design is based on the turn signal, acceleration pedal, steering, steering wheel gradient, brake, and trailer switch which should not be considered at this stage.) |
| FN-2 | The LKA feature shall continuously detect and track the driving lanes on the road. |
| FN-3 | The LKA feature shall estimate and anticipate the path the vehicle will take. (Detailed design based on vehicle dynamics, kinematics, lane detection, and path anticipation. This detail yet considered at this stage of the design process.) |
| FN-4 | The LKA feature shall receive a driver input request to turn on the feature. |
| FN-5 | The LKA feature shall receive a driver input request to turn off the feature. |

| # | Requirement |
|---|---|
| FN-6 | The LKA feature shall always provide the ability for driver to manually override the LKA's intended torque overlay using the mechanical connection between the steering wheel and the steering rack. (Design is based on: turn signal, acceleration pedal, steering, steering wheel gradient, brake – which are also not considered at this stage.) |
| FN-7 | The LKA feature shall detect that the driver has no hands on the steering wheel after a certain amount of time. |
| FN-8 | The LKA feature shall notify the driver via a warning chime when the LKA feature detects that the driver has no hands on the steering wheel. |
| FN-9 | The LKA feature shall provide information and warning to the driver about the different states of the LKA feature with the use of an HMI visual, audio, and haptic notifications. |
| FN-10 | The LKA feature shall be disabled if activating the LKA feature along with other features/features places the vehicle and driver in an undesired/unsafe/non-deterministic state. |

## 7.2 LKA FAD

Figure 7.1 shows the FAD created from the Feature-level functional requirements in Section 7.1.

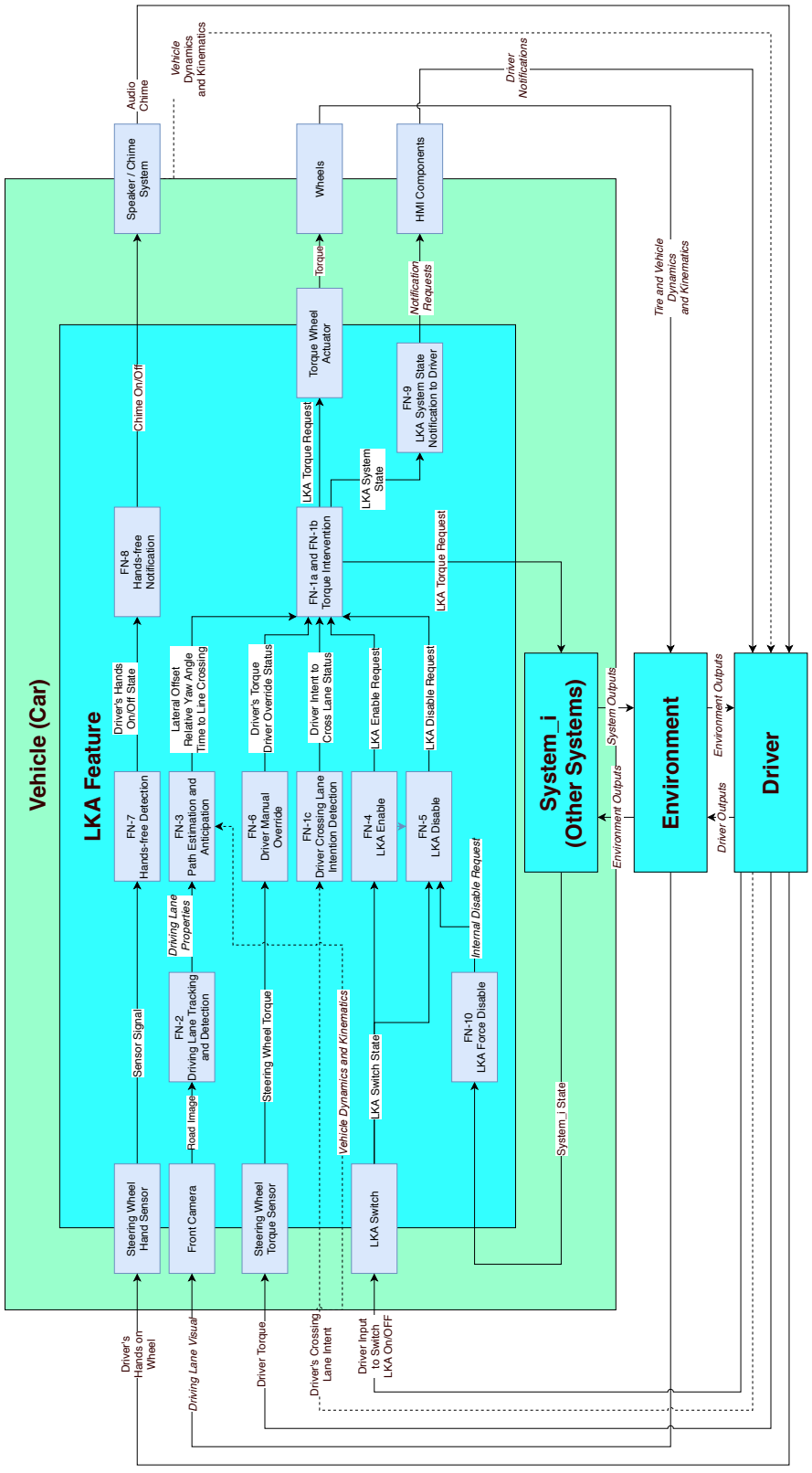



Figure 7.1: FAD of Sample LKA

## 7.3 HAZOP Guide Words

For purposes of this application, the basic hazard guide words stated in Section 6.3.1 are to be used.

## 7.4 Badness and Failure Analysis

This section documents the results of performing the badness and failure analysis in accordance with the Section 6.4.

### 7.4.1 FFA

Figure 7.2 shows the FFA by applying the hazard guide words in Section 7.3 on the LKA Feature-level Functional Requirements in Section 7.1.

The FFA analysis was performed and the resulting table is documented in accordance with Section 6.4.1.

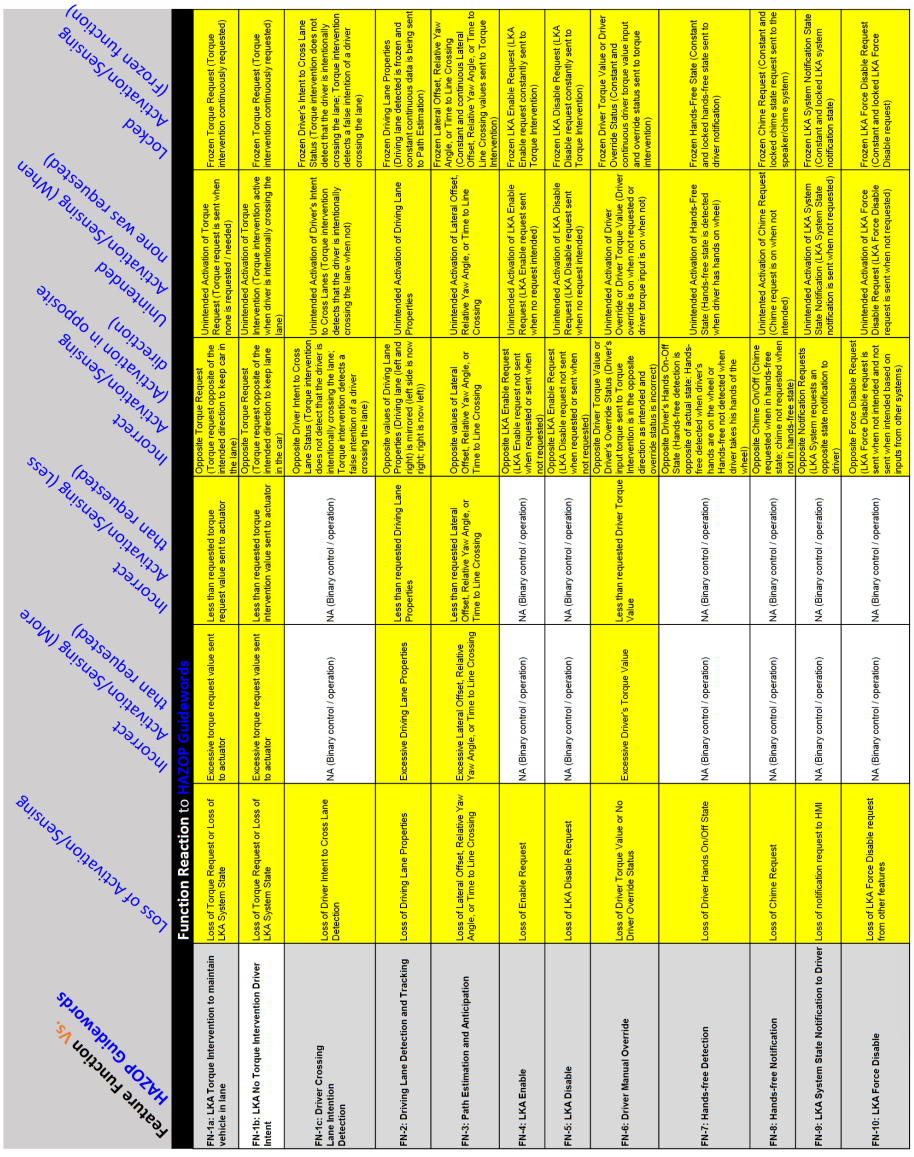



Figure 7.2: FFA of Sample LKA

### 7.4.2 BTA

BTA was applied to the functions in LKA FAD referenced in Section 7.2.

It was observed that most of the functions in LKA show directly-proportional effect to badness transfer such that each input badness driven by a hazard guide word produces the same hazard guide word reaction type on the output.

For example, *Loss of Enable Request* from FN-4 yields a *Loss of Torque Value Request* at FN-1a&b. Most BTA results can then be derived directly from the FFA table. As such, the intermediate BTA tables are not documented.

LKA Functions that show inversely-proportional effect are not missed. These are also considered and included in the overall BTA analysis.

The final BTA is incorporated into the BNT analysis in Figure 7.4.3.

### 7.4.3 FFA + BTA + FAD = BNT

BNT analysis was performed on the LKA Feature concept synthesizing the information and results from the FFA in Section 7.4.1 and BTA in Section 7.4.2 against the FAD in Section 7.2.

Figure 7.4.3 shows the results of the BNT analysis. The BNT analysis was performed and the resulting table is documented in accordance with Section 6.4.3.

| | Function Level Failure or Malfunction | | Feature Reactions | |
|---|---|---|---|---|
| **HAZOP Guideword** | FN-7 | FN-8 | **FN-8** | |
| Loss of Activation/Sensing | Loss of Driver Hands On/Off State | Loss of Audible Warning | The LKA feature is unable to notify the driver via an audible warning when the driver has no hands on the steering wheel. | |
| Incorrect Activation/Sensing (More than requested) | NA (Binary control / operation) | NA (Binary control / operation) | NA (Binary control / operation) | |
| Incorrect Activation/Sensing (Less than requested) | NA (Binary control / operation) | NA (Binary control / operation) | NA (Binary control / operation) | |
| Incorrect Activation/Sensing (Activation in opposite direction) | Opposite Driver's Hands On-Off State (Hands-free detection is opposite of actual state: Hands-free detected when driver's hands are on the wheel or Hands-free not detected when driver takes hands off the wheel) | Opposite warning On/ Off (warning requested when in hands-free state; warning not requested when not in hands-free state) | The LKA feature incorrectly notifies the driver via an audible warning when the driver has hands on the steering wheel. | The LKA feature incorrectly does not notify the driver via an audible warning when the driver has no hands on the steering wheel. |
| Unintended Activation/Sensing (When none was requested) | Unintended Activation of Hands-Free State (Hands-free state is detected when driver has hands on wheel) | Unintented Activation of Audible Warning Request (Audible warning request is on when not intended) | The LKA feature incorrectly notifies the driver via an audible warning when the driver has hands on the steering wheel. | |
| Locked Activation/Sensing (Frozen function) | Frozen Hands-Free State (Constant and locked hands-free state sent to driver notification) | Frozen Warning Request (Constant and locked warning state request sent to the warning system) | The LKA feature continuously notifies the driver that the driver's hands are not on the steering wheel. | |

| | Function Level Failure or Malfunction | | | | | | | | | Feature Reactions | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **HAZOP Guideword** | FN-2 | FN-3 | FN-6 | FN-1c | FN-4 | FN-10 | FN-5 | FN-1a&b | FN-9 | **FN1a** | **FN-1b** | **FN-9** |
| Loss of Activation/ Sensing | Loss of Driving Lane Properties | Loss of Lateral Offset, Relative Yaw Angle, or Time to Line Crossing | -> | -> | -> | -> | -> | Loss of Torque Request or Loss of LKA Feature State | Loss of notification of correct request to HMI | When the LKA feature and car are on and the driver is crossing the lane marking unintentionally, the LKA feature DOES NOT provide torque intervention to maintain the vehicle in its driving lane by moving the vehicle laterally as intended – usually in direction of the lane center. | The LKA feature INTERVENES if the driver is crossing the lane marking intentionally. | The LKA feature DOES NOT provide correct information and warning to the driver about the different states of the LKA feature with the use of an HMI visual, audio, and haptic notifications. |
| | | Loss of Lateral Offset, Relative Yaw Angle, or Time to Line Crossing | -> | -> | -> | -> | -> | | | | | |
| | | | Loss of Driver Torque Value or No Driver Override Status | -> | -> | -> | -> | Unintended Torque Request or Loss of Feature State | | | | |
| | | | | Loss of Driver Intent to Cross Lane Detection | -> | -> | -> | | | | | |
| | | | | | Loss of Enable Request | -> | -> | Loss of Torque Request or Loss of LKA Feature State | | | | |
| | | | | | | Loss of LKA Force Disable request from other features | Loss of LKA Disable Request | Unintended Torque Request or Loss of LKA Feature State | | | | |
| | | | | | | | Loss of LKA Disable Request | | | | | |
| | | | | | | | | Loss of LKA Feature State | | | | |
| | | | | | | | | Loss of Torque Request | -> | | | -> |
| | | | | | | | | | Loss of notification request to HMI | -> | -> | The LKA feature DOES NOT provide correct information and warning to the driver about the different states of the LKA feature with the use of an HMI visual, audio, and haptic notifications. |

| | Function Level Failure or Malfunction | | | | | | | | | Feature Reactions | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **HAZOP Guideword** | FN-2 | FN-3 | FN-6 | FN-1c | FN-4 | FN-10 | FN-5 | FN-1a&b | FN-9 | **FN1a** | **FN-1b** | **FN-9** |
| Incorrect Activation/ Sensing (More than requested) | Excessive Driving Lane Properties (Driving lane value received by Path Estimation and Anticipation is more than actual) | Excessive Lateral Offset, Relative Yaw Angle, or Time to Line Crossing | -> | -> | -> | -> | -> | Excessive torque request value sent to actuator | NA (Binary control / operation) | When the LKA feature and car are on and the driver is crossing the lane marking unintentionally, the LKA feature provides EXCESSIVE torque intervention to maintain the vehicle in its driving lane by moving the vehicle laterally as intended – usually in direction of the lane center. | | |
| | | Excessive Lateral Offset, Relative Yaw Angle, or Time to Line Crossing | -> | -> | -> | -> | -> | | | | | |
| | | | Excessive Driver Torque Value | -> | -> | -> | -> | Less torque value sent to actuator | | When the LKA feature and car are on and the driver is crossing the lane marking unintentionally, the LKA feature DOES NOT PROVIDE ENOUGH torque intervention to maintain the vehicle in its driving lane by moving the vehicle laterally as intended – usually in direction of the lane center. | | |
| | | | | NA (Binary control / operation) | -> | -> | -> | N/A | | N/A | | |
| | | | | | NA (Binary control / operation) | -> | -> | | | | | |
| | | | | | | NA (Binary control / operation) | NA (Binary control / operation) | | | | | |
| | | | | | | | NA (Binary control / operation) | | | | | |
| | | | | | | | | Excessive torque request value sent to actuator | | When the LKA feature and car are on and the driver is crossing the lane marking unintentionally, the LKA feature provides EXCESSIVE torque | | |

| | Function Level Failure or Malfunction | | | | | | | | | Feature Reactions | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **HAZOP Guideword** | FN-2 | FN-3 | FN-6 | FN-1c | FN-4 | FN-10 | FN-5 | FN-1a&b | FN-9 | **FN1a** | **FN-1b** | **FN-9** |
| Incorrect Activation/ Sensing (Less than requested) | Less than requested Driving Lane Properties received by Path Estimation and Anticipation | Less than requested Lateral Offset, Relative Yaw Angle, or Time to Line Crossing | -> | -> | -> | -> | -> | Less than requested torque request value sent to actuator | NA (Binary control / operation) | When the LKA feature and car are on and the driver is crossing the lane marking unintentionally, the LKA feature DOES NOT PROVIDE ENOUGH torque intervention to maintain the vehicle in its driving lane by moving the vehicle laterally as intended – usually in direction of the lane center. | | |
| | | Less than requested Lateral Offset, Relative Yaw Angle, or Time to Line Crossing | -> | -> | -> | -> | -> | | | | | |
| | | | Less than requested Driver Torque Value | -> | -> | -> | -> | Excessive torque request value sent to actuator | | When the LKA feature and car are on and the driver is crossing the lane marking unintentionally, the LKA feature provides EXCESSIVE torque intervention to maintain the vehicle in its driving lane by moving the vehicle laterally as intended – usually in direction of the lane center. | | |
| | | | | NA (Binary control / operation) | -> | -> | -> | N/A | | N/A | | |
| | | | | | NA (Binary control / operation) | -> | -> | | | | | |
| | | | | | | NA (Binary control / operation) | NA (Binary control / operation) | | | | | |
| | | | | | | | NA (Binary control / operation) | | | | | |

| | Function Level Failure or Malfunction | | | | | | | | Feature Reactions | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **HAZOP Guideword** | FN-2 | FN-3 | FN-6 | FN-1c | FN-4 | FN-10 | FN-5 | FN-1a&b | FN-9 | **FN1a** | **FN-1b** | **FN-9** |
| Incorrect Activation/ Sensing (Activation in opposite direction) | Opposite values of Driving Lane Properties (Driving lane (left and right) is mirrored (left side is now right; right is now left)) | Opposite values of Lateral Offset, Relative Yaw Angle, or Time to Line Crossing | -> | -> | -> | -> | -> | Opposite Torque Value Request (Torque request opposite of the intended direction to keep lane in the car) | Opposite Notification Requests (LKA Feature requests an opposite state notification to driver) | When the LKA feature and car are on and the driver is crossing the lane marking unintentionally, the LKA feature provides torque intervention in the OPPOSITE DIRECTION of the center of the road. | The LKA feature INCORRECTLY INTERVENES if the driver is crossing the lane marking intentionally or DOES NOT INTERVENE when the driver is crossing the lane marking unintentionally. | The LKA feature provides INCORRECT information and warning to the driver about the different states of the LKA feature with the use of HMI visual, audio, and haptic notifications. |
| | | Opposite values of Lateral Offset, Relative Yaw Angle, or Time to Line Crossing | -> | -> | -> | -> | -> | | | | | |
| | | | Opposite Driver Torque Value or Driver's Override Status (Driver's input torque sent to Torque Intervention is in the opposite direction as intended and override status is incorrect) | -> | -> | -> | -> | | | | | |
| | | | | Opposite Driver Intent to Cross Lane Status (Torque intervention does not detect that the driver is intentionally crossing the lane; Torque intervention detects a false intention of a driver crossing the lane) | -> | -> | -> | | | | | |
| | | | | | Opposite LKA Enable Request (LKA Enable request not sent when requested or sent when not requested) | -> | -> | | | | | |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | Opposite Force Disable Request (LKA Force Disable request is sent when not intended and not sent when intended based on inputs from other systems) | Opposite LKA Enable Request (LKA Disable request not sent when requested or sent when not requested) | | | | | |
| | | | | | | | Opposite LKA Enable Request (LKA Disable request not sent when requested or sent when not requested) | | | | | |
| | | | | | | | | Opposite LKA Feature State | | | | |
| | | | | | | | | Opposite Torque Value Request (Torque request opposite of the intended direction to keep lane in the car) | -> | | | -> |
| | | | | | | | | | Opposite Notification Requests (LKA System requests an opposite state notification to driver) | -> | -> | The LKA feature provides INCORRECT information and warning to the driver about the different states of the LKA feature with the use of an HMI visual, audio, and haptic notifications. |

| | Function Level Failure or Malfunction | | | | | | | | | Feature Reactions | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **HAZOP Guideword** | FN-2 | FN-3 | FN-6 | FN-1c | FN-4 | FN-10 | FN-5 | FN-1a&b | FN-9 | **FN1a** | **FN-1b** | **FN-9** |
| Unintended Activation/ Sensing (When none was requested) | Unintended Activation of Driving Lane Properties | Unintended Activation of Lateral Offset, Relative Yaw Angle, or Time to Line Crossing | -> | -> | -> | -> | -> | Unintended Activation of Torque Request or LKA System State (Torque request is sent when none is requested / needed) | Unintended Activation of LKA System State Notification (LKA System State notification is sent when not requested) | When the LKA feature or the car is OFF or the driver is crossing the lane marking intentionally, the LKA feature INCORRECTLY provides torque intervention when none is required. | The LKA feature incorrectly DOES NOT intervene when the driver is crossing the lane marking unintentionally. | The LKA feature provides information and warning to the driver about the different states of the LKA feature with the use of HMI visual, audio, and haptic notifications when NONE was requested. |
| | | Unintended Activation of Lateral Offset, Relative Yaw Angle, or Time to Line Crossing | -> | -> | -> | -> | -> | | | | | |
| | | | Unintended Activation of Driver Override or Driver Torque Value (Driver override is on when not requested or driver torque input is on when not) | -> | -> | -> | -> | Unintended DEACTIVATION of Torque Request or LKA System State (Torque request is NOT sent when it is requested / needed) | | | | |
| | | | | Unintended Activation of Driver's Intent to Cross Lanes (Torque intervention detects that the driver is intentionally crossing the lane when not) | -> | -> | -> | | | | | |
| | | | | | Unintended Activation of LKA Enable Request (LKA Enable request sent when no request intended) | -> | -> | Unintended Activation of Torque Request or LKA System State (Torque request is sent when none is requested / needed) | | | | |
| | | | | | | Unintended Activation of LKA Force Disable Request (LKA Force Disable request is sent when not requested) | Unintended Activation of LKA Disable Request (LKA Disable request sent when no request intended) | Unintended DEACTIVATION of Torque Request or LKA System State (Torque request is NOT sent when it is requested / needed) | | | | |
| | | | | | | | Unintended Activation of LKA Disable Request (LKA Disable request sent when no request intended) | | | | | |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | Unintended Activation of LKA System State | | | |
| | | | | | | | | Unintended Activation of Torque Request (Torque request is sent when none is requested / needed) | -> | | -> |
| | | | | | | | | | Unintended Activation of LKA System State Notification (LKA System State notification is sent when not requested) | -> | -> | The LKA feature provides information and warning to the driver about the different states of the LKA feature with the use of an HMI visual, audio, and haptic notifications when NONE was requested. |

| | Function Level Failure or Malfunction | | | | | | | | | Feature Reactions | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **HAZOP Guideword** | FN-2 | FN-3 | FN-6 | FN-1c | FN-4 | FN-10 | FN-5 | FN-1a&b | FN-9 | **FN1a** | **FN-1b** | **FN-9** |
| Locked Activation/ Sensing (Frozen function) | Frozen Driving Lane Properties (Driving lane detected is frozen and constant continuous data is being sent to Path Estimation) | Frozen Lateral Offset, Relative Yaw Angle, or Time to Line Crossing (Constant and continuous Lateral Offset, Relative Yaw Angle, or Time to Line Crossing values sent to Torque Intervention ) | -> | -> | -> | -> | -> | Frozen Torque Request value or LKA System State (Torque intervention continuously requested) | Frozen LKA System Notification State (Constant and locked LKA system notification state) | The LKA feature continuously provides torque intervention with constant value and unresponsive to stimulus. | The LKA feature is disabled and unresponsive to stimulus. | The LKA feature provides constant (frozen/locked) value of information and feature state to the driver regarding the LKA feature with the use of HMI visual, audio, and haptic notifications. It is unresponsive to stimulus. |
| | | Frozen Lateral Offset, Relative Yaw Angle, or Time to Line Crossing (Constant and continuous Lateral Offset, Relative Yaw Angle, or Time to Line Crossing values sent to Torque Intervention ) | -> | -> | -> | -> | -> | | | | | |
| | | | Frozen Driver Torque Value or Driver Override Status (Constant and continuous driver torque value input and override status sent to torque intervention) | -> | -> | -> | -> | | | | | |
| | | | | Frozen Driver's Intent to Cross Lane Status (Torque intervention does not detect that the driver is intentionally crossing the lane; Torque intervention detects a false intention of a driver crossing the lane) | -> | -> | -> | | | | | |

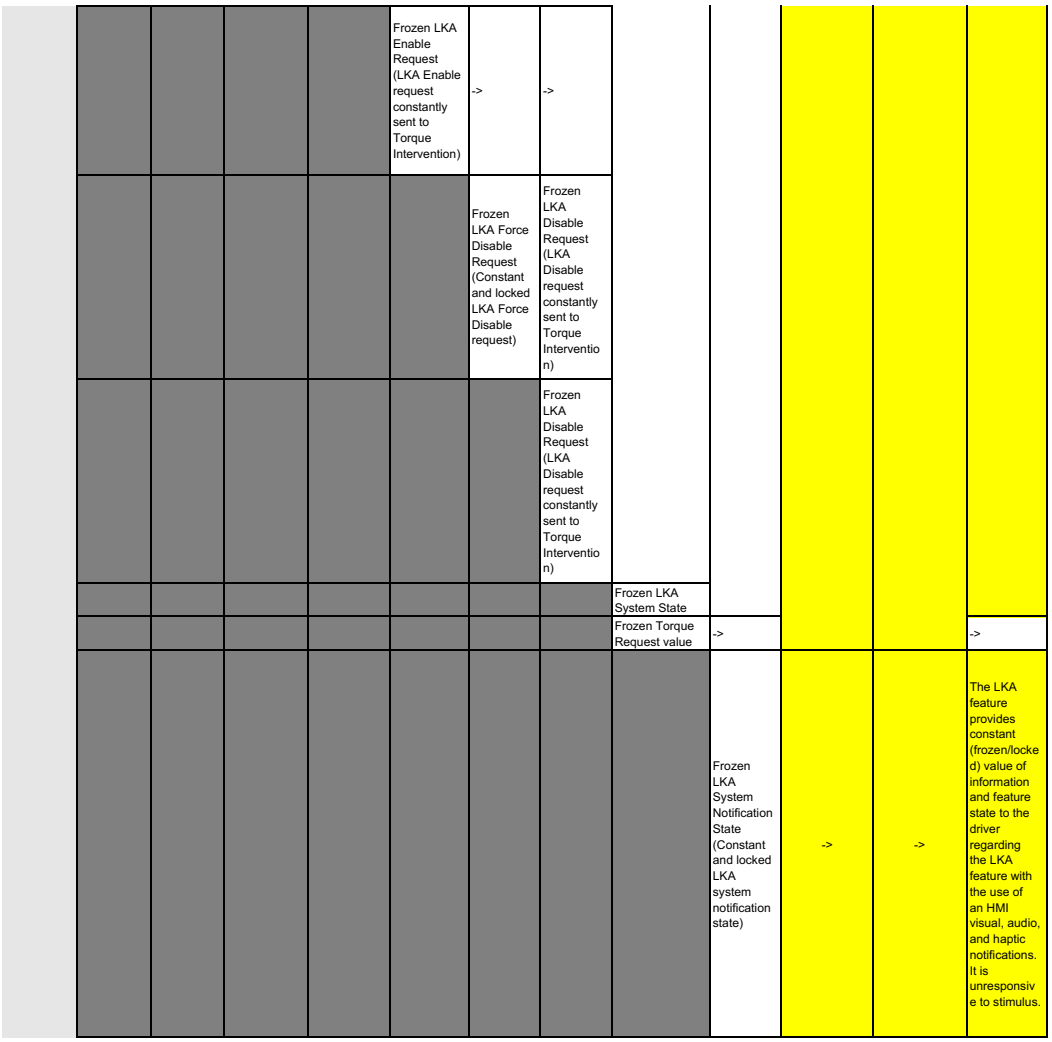



Figure 7.3: BNT w/ Feature Reaction Against Requirements of Sample LKA

## 7.5 Global Badness to Requirements Failure

The Feature-level reactions are extended from the BNT analysis as shown in Figure 7.4.3 indicated by the Feature Reactions column label.

These Feature-level reactions are obtained in accordance with the process defined in Section 6.5.

The reactions are described such that they are a direct translation of requirement failures observed at the boundary functions.

# Chapter 8

# Incremental HAZOP$^{+}$

This section describes a relatively detailed overview of an addition to the proposed approach that facilitates the development of incremental system safety assurance cases.

As discussed in Section 4.2, to provide an efficient HAZOP process, the functional depth of abstraction should not be fixed, but rather based upon functional decomposition.

An incremental HAZOP$^{+}$, which we will call HAZOP$^{\Delta}$, can be developed over the process of functional decomposition performed over Goal Structuring Notation -inspired (GSN) diagrams referencing the different levels of FADs.

The remainder of this chapter presents, by way of an example, the proposed extension of the systematic HAZOP$^{+}$ process performed over functional decomposition.

## 8.1 LKA Requirement Decomposition over Functional Architecture Diagrams

This chapter describes a high-level but detailed Functional Architecture (FA) for the LKA Feature. This FA is presented as a hierarchy of FADs, which implements top LKA functional requirements in a modular way and model LKA's control structure. The structure of the hierarchy is shown in Figure 8.1, where green rectangles refer to FADs that model LKA with an increased level of detail from the root of the hierarchy to its leaves. Each step in this hierarchy implements a requirement, and grey rectangles encompassing green rectangles give a suggestive name to the corresponding RDstep. This is discussed in full detail below.

### 8.1.1 General Schema of Requirement Decomposition (RD)

The RD process performed over the associated hierarchy of FADs modelling system $X$ is a pre-requisite to $X$'s HAZOP$^+$.

A typical RDstep takes a functional requirement $R$ for a system $X$ considered as a function $F$ as its input, and produces a FAD consisting of $n$ functional blocks $F_1....F_n$ together with their functional requirements $R_1...R_n$; written as $FAD(R) = (F_1, ..., F_n)$ for the FAD, and $F_i \models R_i$, $i = 1...n$ for satisfiability relations between functions and functional requirements. It is assumed that if component functions satisfy their functional requirements, $F_i \models R_i$, then the decomposed function $F$ satisfies its functional requirement, $F \models R$. Then it is said that $FAD(R)$ (together with functional requirements $R_i$) *implements*

$R$. Also admitted functional requirements spanning several blocks in $FAD(R)$, *e.g.*, functional requirement $R_{ijk}$ for the composition $F_{ijk}$ of three blocks $F_i, F_j, F_k$. This means that block $F_{ijk}$ is implicitly present in $FAD(R)$ but is already functionally decomposed without the corresponding functional requirement decomposition - the latter will be developed at the next step. Such "early" functional decomposition may help to understand the FAD and guides its future RD.

For another functional requirement on $F$, given $R'$, one should provide another decomposition $FAD(R') = F'_1....F'_{n'}$. An RDstep can also be purely logical, when functional requirement $R$ is decomposed into a conjunction of more detailed functional requirements $R_1...R_n$ for $F$ without its functional decomposition: $(F \models R_i)_{i=1...n}$ implies $F \models R$.

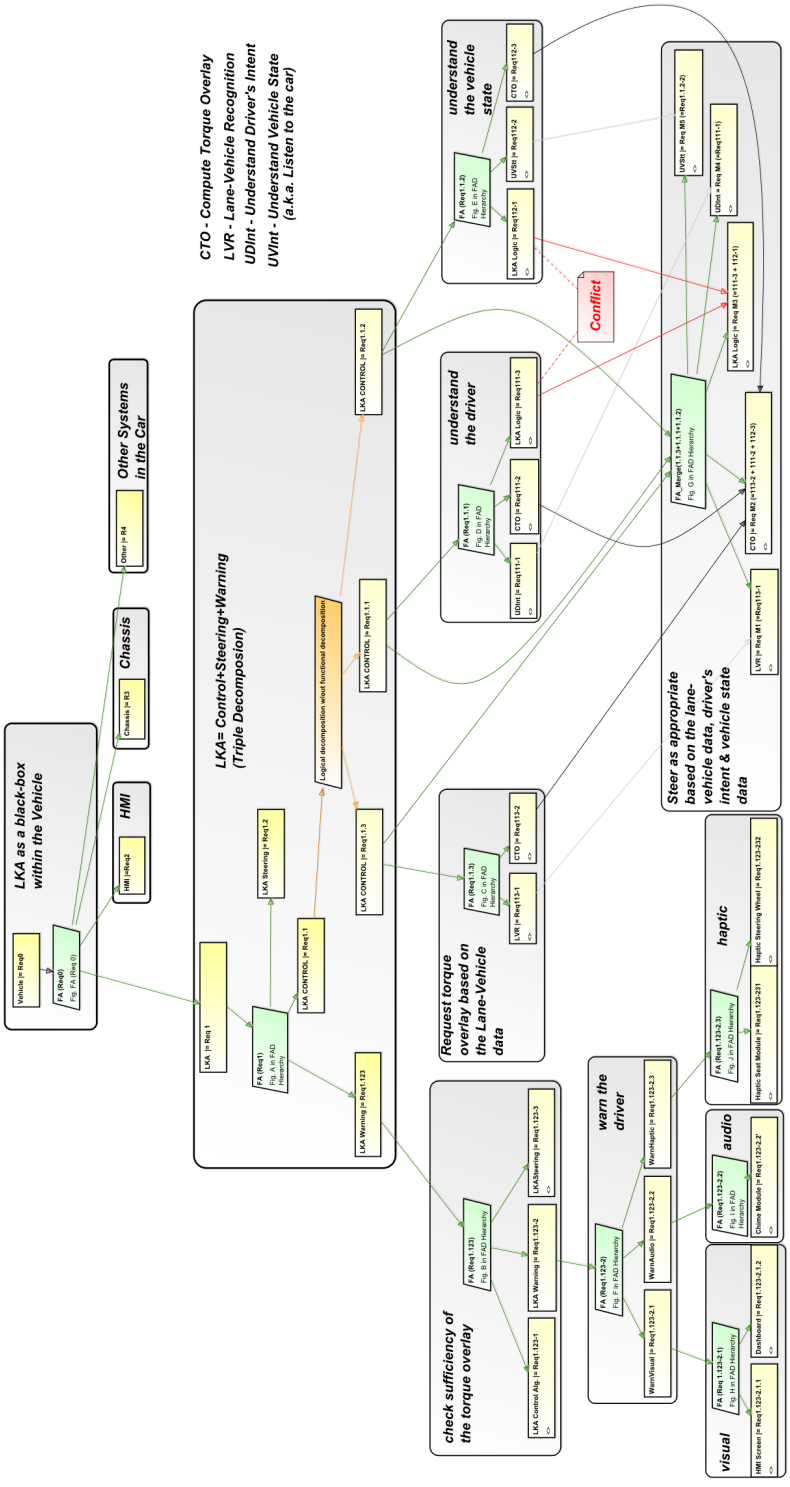



Figure 8.1: GSN-inspired Diagram of LKA Requirement Decomposition

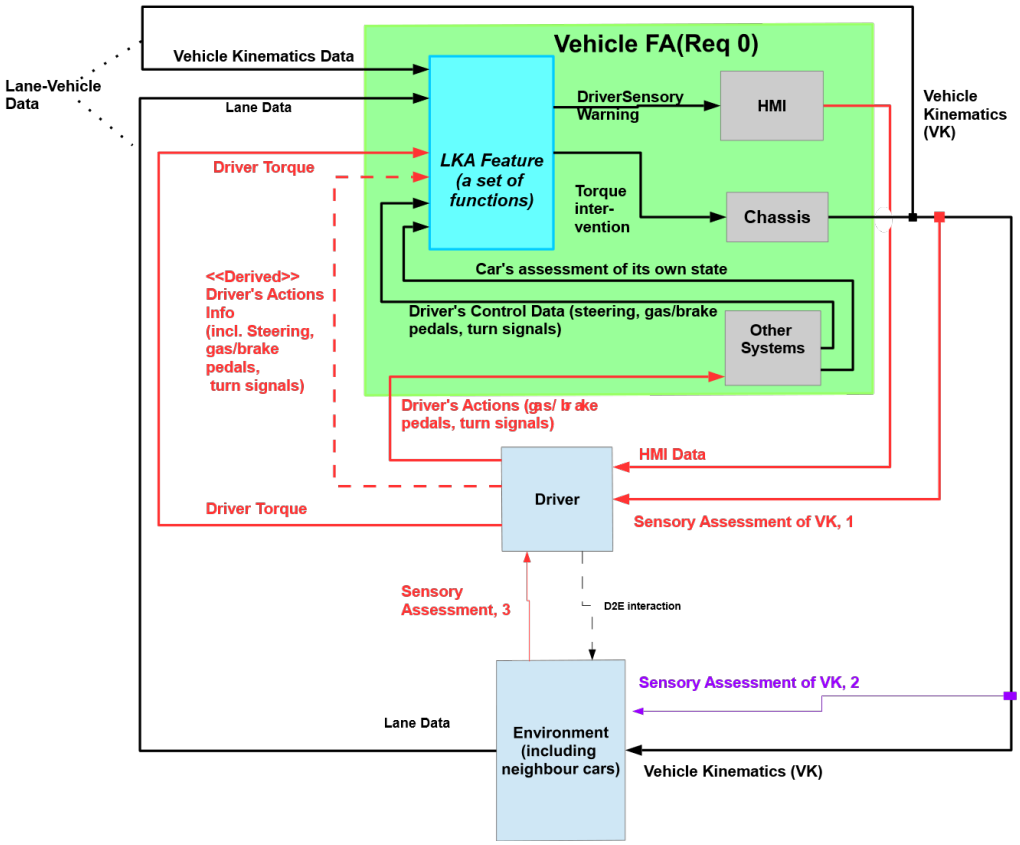



Figure 8.2: FA (Req 0)

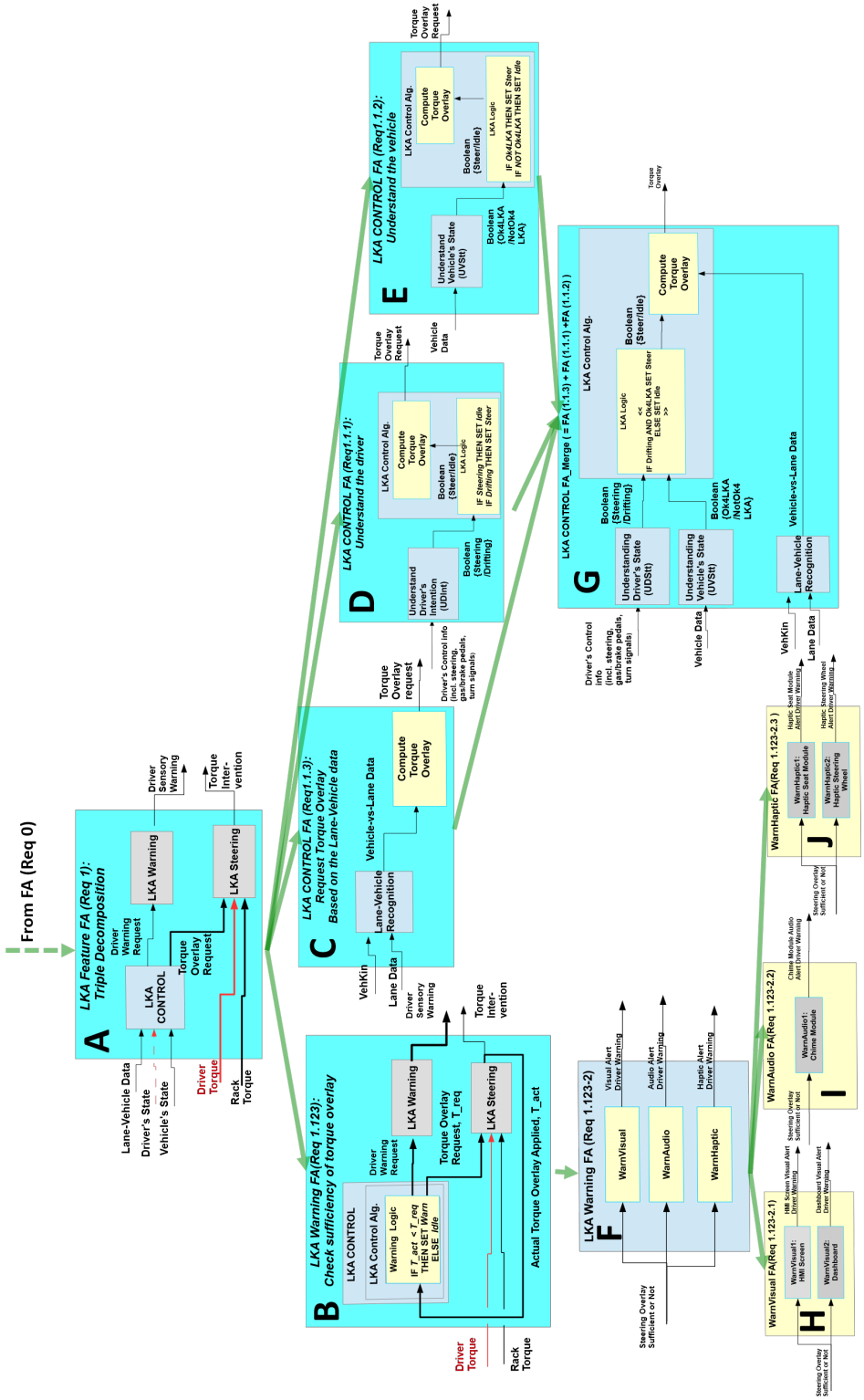



Figure 8.3: Hierarchy of LKA FADs

### 8.1.2 Initial Top-Level Requirement and Model

The RD process begins with defining the vehicle-level requirement along with the consideration of the most abstract model. In this case, the scope is defined between the vehicle, driver, and environment

The requirement imposed on the vehicle at the top-level abstraction is:

- *Req_0*: The Vehicle shall actively assist the driver in keeping the vehicle inside the driving lane in an intelligent way based on the understanding of the vehicle itself and the driver's intention.

### 8.1.3 Vehicle-level Abstraction

The next step is to perform RD on the initial requirement *Req_0* on the Vehicle shown in the FAD in Figure 8.2 and referenced in the top node of the GSN-inspired diagram (Figure 8.1), "LKA as a black-box within the Vehicle."

The interactions between the elements of the model are represented by directed arrows. The labels attached to the arrows represent the transmitted signals. As shown, the LKA Feature interacts with other Vehicle elements, Driver, and Environment.

*Req_0* is then decomposed into the following requirements:

- *R1*: The LKA Feature shall actively assist the driver in keeping the vehicle inside the driving lane in an intelligent way based on the understanding of the vehicle itself and the driver's intention; the LKA Feature shall also warn the driver if it cannot fully perform the requirement.

- *R2*: Human-Machine Interface shall assist the LKA in the driver's understanding of the vehicle.

- *R3*: Chassis shall move the vehicle.
- *R4*: Other Systems in the car shall assist in moving the vehicle.

The vehicle in FAD (Figure 8.2) and the corresponding FA (Req 0) node of the GSN-inspired diagram (Figure 8.1) show the hierarchical decomposition of the vehicle into the following elements to satisfy the requirements above:

- LKA = Control + Steering + Warning satisfies *R1*.
- Human-Machine Interface satisfies *R2*.
- Chassis satisfies *R3*.
- Other Systems in the car satisfies *R4*.

### 8.1.4 First-Level Decomposition of the LKA Feature

After the decomposition of the vehicle-level requirement *Req_0* and the definition of the initial elements of the vehicle, the next step in the hierarchy is to perform RD on Requirement *R1*.

The following state the elements of the triple decomposition of Requirement *R1* shown on Element A - Hierarchy of LKA FADs in Figure 8.3 and referenced in the GSN-inspired diagram (FA (Req 1) in Figure 8.1):

- *Req1.1*: LKA Control and LKA Steering shall provide torque intervention to steer the car appropriately based on understanding the driver's intent, the state of the vehicle, and lane-tracking.
- *Req1.2*: LKA Steering shall provide torque intervention data to LKA Control; based on this data, LKA Control shall check if the actual torque

intervention is less than requested and provide the corresponding signal to LKA Warning; based on this signal, LKA Warning shall warn the driver if and only if the signal is active.

### 8.1.5 Second-Level Decomposition of the LKA Feature

Elements C, D, and E - Hierarchy of LKA FADS in Figure 8.3 show the RD of Requirement *R1.1* as per their respective elements in the GSN-inspired diagram (Figure 8.1). Note that for *R1.2*, the RD process only considered pure logical decomposition as opposed to also considering its functional decomposition. The following are the results of the logical decomposition of *R1.1*:

- *Req1.1.1 (Element D)*: UDInt shall identify driver's intent; based on this data, LKA Logic shall determine if torque should be applied or not; based on this data, CTO shall compute the required torque overlay and send the request.

- *Req1.1.2 (Element E)*: UDVStt shall identify the state of the vehicle; based on this data, LKA Logic shall determine if torque should be applied or not; based on this data, CTO shall compute the required torque overlay and send the request.

- *Req1.1.3 (Element C)*: Lane-Vehicle-Recognition (LVR) shall recognize the driving lane; based on this data, based on this data, CTO shall compute the required torque overlay and send the request.

A special focus is placed on *Req1.2*. This requirement spans multiple components of the LKA Feature. LKA Control, Steering, and Warning all have pieces of the requirement attributed to them. Element B - Hierarchy of

LKA FADS in Figure 8.3 show the FAD for *R1.123* which is a restatement of requirement *R1.2*. No true decomposition was performed at this step, but the requirements for *R1.123-1*, *R1.123-2*, and *R1.123-3* are obtained piecemeal according to *R1.123*:

- *R1.123-1*: LKA Steering shall provide torque intervention data to LKA Control.
- *R1.123-2*: LKA Control shall check if the actual torque intervention is less than requested and provide the corresponding signal to LKA Warning.
- *R1.123-3*: LKA Warning shall warn the driver if and only if the signal from LKA Control is active

### 8.1.6 Third-Level Decomposition of the LKA Feature

The next level of RD involves the *merge* of requirements *Req1.1.1*, *Req1.1.2*, and *Req1.1.3*.

The GSN-inspired diagram (Figure 8.1) shows that merging the CTO requirements do not create a conflict since the coverage of the logic do not overlap.

On the other hand, a conflict exists when merging LKA Logic: *Req111-3* states that if Driver is Drifting, the LKA shall steer the car, while *Req112-3* states that LKA shall not steer the car if it interferes with other critical systems in the car (even if Driver is Drifting).

Symmetrically, *Req112-3* allows LKA to steer if it is okay with other systems, but *Req11-3* shall prohibit this if the Driver is Steering.

In order to consider the two component FADs as two views of one integral FAD, the conflict needs to be resolved and then FADs can be merged as shown by FAD G in Figure 8.3.

The integrated functional requirement for LKA Logic is functional requirement *ReqM3* (see below); functional requirements for other components of the merge FAD G follow straightforwardly from their counterparts before the merge.

- *ReqM1*: LVR shall recognize the driving lane.
- *ReqM2*: CTO Overlay Computing shall either request torque overlay when the command Steer is received or not apply torque overlay when a command to Idle is received from LKA Logic; the torque overlay request value shall be dependent on LVR.
- *ReqM3*: LKA Logic shall only send a Steer command if the driver is in a Drifting state AND it is safe for LKA to apply torque intervention. If the driver is in a Driving state or it is NOT safe for LKA to apply torque intervention, LKA Logic sends an Idle command.
- *ReqM4*: UDInt shall identify the driver's intent to Steer or not steer (Drifting).
- *ReqM5*: UDVStt shall determine if it is either safe for LKA to apply torque intervention or not based on Vehicle Data.

The next piece in the third level of decomposition is the further RD of *R1.123* based on LKA Warning which is shown on Element F - Hierarchy of LKA FADs in Figure 8.3 as FA (Req 1.123-2). The following are the requirements derived from this RD

- *Req1.123-2.1*: WarnVisual shall provide visual warning alerts to the driver.
- *Req1.123-2.2*: WarnAudio shall provide audio warning alerts to the driver.
- *Req1.123-2.3*: WarnHaptic shall provide haptic warning alerts to the driver.

The last step related to the RD of LKA Warning is the decomposition of *Req1.123-2.1*, *Req1.123-2.2*, and *Req1.123-2.3*. which are allocated to and satisfied by the functional components shown on Elements H, I, and J - Hierarchy of LKA FADS, respectively.

## 8.2 Extending HAZOP$^{\Delta}$

The sections within Section 8.1 describe how HAZOP$^{\Delta}$ differs from HAZOP$^{+}$. To complete HAZOP$^{\Delta}$, we can use the processes defined in Section 6 starting from malfunction analysis in subsection 6.4 all the way to defining the vehicle-level hazardous events.

# Chapter 9

# Benefits

## 9.1 HAZOP$^+$ vs. Traditional HAZOP

**Traditional HAZOP**

Execution of the PHA-level HAZOP on LKA (Chapter 2) as per typical industry practice results in Figure 4.3 in Section 4.6.

Figure 4.3 arrives at the Feature-level reactions based on the HAZOP team's heuristics, experience, and expertise as discussed in Section 2.2.2.

**HAZOP$^+$**

Execution of the proposed systematic approach HAZOP$^+$(Chapter 6) on LKA (Chapter 7) results in Figure 7.4.3 in Section 7.4.3.

Figure 7.4.3 arrives at the Feature-level reactions based on a systematic process by first defining the Feature-level functional requirements (Section 7.1) formally.

Then, the FAD was created (Section 7.2) to show the functional elements and their dependencies against the requirements.

Next, the Badness and Failure Analysis (FFA, BTA, and BNT) was performed

on the Feature-level Functional Requirements against the hazard guide words (Section 7.3) while traversing the FAD. The detail of the analysis is described in Section 7.4.

Finally, the results of the Badness and Failure Analysis have been extended to the Feature-level hazards, which are the violation of the Feature-level Functional Requirements. The details of the analysis is described in Section 7.5.

**Summary of Comparison**

HAZOP$^+$ shows a methodical and traceable approach to HAZOP over a representative industry method.

Traceability can be formed between the intermediate and final results, intermediate steps, and initial Feature-level functional requirements.

The analyses in the proposed approach are performed objectively which eliminates the subjective nature of the heuristic and expert-based approach currently practised in industry.

The proposed approach also shows the relationships and dependencies not only between the functional components of the feature, but also between the components of the Car-Driver-Environment-Feature model.

## 9.2 The potential for tool support and automation

The extension of the proposed systematic HAZOP$^+$ to the incremental HAZOP$^+$ analysis consists of the Functional Requirement Decomposition (Section 8.1) over GSN-inspired diagrams and FAD, Hazard Guide Words (Section 6.3), FFA (Section 6.4.1), BTA (Section 6.4.2), BNT (Section 6.4.3), and extending the badnesses to requirements failure (Section 6.5).

A software tool can be developed to unify and reference all the products

of these processes such that change propagation can be streamlined and the scope of its impact be assessed more reliably.

The GSN-inspired diagrams will contain references to the FADs. The relevant requirements and functions that satisfy them are traceable within the software tool, such that a malfunction at any given level of abstraction can link directly back to the requirement that is failed.

Developing baselines and the base safety case will be maintained easily - any modifications to the base safety case can then be propagated to different versions of the product.

## 9.3 Benefits of model-based documentation to incremental safety case construction and management

This subsection describes the benefits of HAZOP$^+$to incremental HAZOP$^+$ processes to incremental safety case construction and management.

### 9.3.1 Verifiability and Validatability - Impact Analysis and Traceability

An incremental safety case process that promotes proper scope management is needed to reduce the effort in verification and validation (V&V). The presented incremental HAZOP$^+$ produces GSN-inspired diagrams and FADs, as well as FFAs, BTAs, and BNT tables that reference them. Traceability is inherent in incremental HAZOP$^+$ and the dynamic levels of abstraction based on the

functional decomposition reduce the effort in V&V activities as the analysis is not required to be performed at the atomic level. The impact is defined and limited to the exact level of abstraction in the decomposition tree.

### 9.3.2 Modularity for Delta Work

Modularity is promoted through the use of GSN-inspired and FA diagrams as dependencies are explicitly shown. As incremental HAZOP$^+$ incorporates dynamic levels of abstraction, change can be made at any level in the decomposition tree. The scope of the impact can be visually assessed and can be verified by traversing the tree. The unaffected branches are then assumed to still be functionally correct.

# Chapter 10

# Future Work

The proposed software tool described in Section 9.2 can be developed and the hazards extended to hazardous events automatically. The current version of HAZOP$^+$ still relies on human input in extending the combination of hazards and operating conditions to hazardous events. There are countless operating conditions that have to be considered in performing HAZOP. Considering which operating conditions will lead to a significant hazardous event is likely to remain a manual process that relies heavily on experience and expert opinion. Furthermore, the use of GSN-inspired and FADs in developing HAZOP and incremental assurance may require more accurate and granular models than what has been presented. More refined and decomposed models that consider full interactions between the components of the vehicle, driver, and environment are required for more accurate and meaningful analyses.

Incremental HAZOP$^+$and its artefacts (FFA, BTA, BNT, and their relationships with FADs and functional decomposition) can be represented and integrated with more defined notation such as the *Failure Propagation and Transformation Notation* (FPTN) in combination with modular representation and compositional

analysis of a system's hardware and software components with *Fault Propagation and Transformation Calculus* (FPTC) [15, 16].

# Chapter 11

# Conclusion

This thesis has presented a systematic, formal, and explicit approach to Hazard and Operability Study (HAZOP$^+$) that aims to address the need for a traceable, explicit, documentable, verifiable, and validatable process, that facilitates the systematic development of incremental safety assurance cases.

A Preliminary Hazard Analysis (PHA) level HAZOP applied to the automotive feature Lane Keeping Assist (LKA), was used as an example for analysis and discussion. A typical approach to HAZOP was used both as a base for, and as a comparative reference to the model-based process framework presented in this thesis and to make apparent the advantages and differences between the proposed framework and current HAZOP approaches.

The initial development life-cycle of the LKA feature was used as an example to demonstrate performing HAZOP at the PHA phase with the intention of developing a reusable HAZOP$^+$process framework that can be applied in the later functional and safety design phases of feature development.

From feature-level requirements definition and initial concept, to concept design and vehicle-level hazards identification, this thesis used a formal and

explicit approach using Functional Architecture Diagrams (FAD) and tabular expressions. This allowed for explicit, unambiguous, traceable, and formal statements and statements of relationships of and between requirements, functional design components, function level malfunctions/reactions, and resulting feature, system, and vehicle level reactions and hazards. This was extended to incremental HAZOP$^+$ which performs Requirement Decomposition over Goal Structuring Notation diagrams and FADs.

The proposed process framework has been shown to add meaningful and logical traceability to what is currently an implicit heuristics-based and expert-driven process.

The modular and explicit nature of the proposed incremental HAZOP$^+$ approach, hazard mitigation design, change-impact analysis, and incremental safety assurance performed on modifications applied to the original design basis at any functional level results in a more traceable, documentable, verifiable, validatable, manageable, and less resource intensive execution.

# Bibliography

[1] T. H. Hubing. "Autonomous Vehicles Will Transform the Field of Automotive EMC". In: *2018 IEEE Symposium on Electromagnetic Compatibility, Signal Integrity and Power Integrity (EMC, SI PI)*. 2018, pp. 1–29 (cit. on p. 1).

[2] T. Chowdhury et al. "Principles for Systematic Development of an Assurance Case Template from ISO 26262". In: *2017 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. 2017, pp. 69–72 (cit. on pp. 1, 2).

[3] International Standards Organization. *ISO 26262-3: Road Vehicles : Functional Safety - Part 3: Concept Phase*. Second. .Part 3: Concept Phase. ISO, 2018 (cit. on pp. 1, 3).

[4] International Standards Organization. *ISO 26262-6: Road Vehicles : Functional Safety - Part 6: Product development at the software level*. Second. .Part 6: Product development at the software level. ISO, 2018 (cit. on p. 1).

[5] J. Rushby et al. *Understanding and evaluating assurance cases*. Technical Report. SRI International, 2015 (cit. on p. 2).

[6] Zinovy Diskin et al. "Assurance via workflow+ modelling and conformance". In: *CoRR* abs/1912.09912 (2019). arXiv: 1912.09912. URL: http://arxiv.org/abs/1912.09912 (cit. on p. 4).

[7] P. Aoanan et al. *Proprietary: Safety Assurance and Using Model Management to Support It - Deliverable D8: Suggestions on how to integrate aspects of the system/software development and safety processes*. McMaster University, 2019 (cit. on p. 8).

[8] C. Easton. "Identifying safety functions using HAZOP". In: *The IEE Seminar on Methods and Tools for SIL Determinationa 2005 (Ref. No. 2005/10986)*. 2005, pp. 6–6/16 (cit. on pp. 20, 21).

[9] K. Li et al. "HAZOP Study on the CTCS-3 Onboard System". In: *IEEE Transactions on Intelligent Transportation Systems* 16.1 (2015), pp. 162–171. ISSN: 1558-0016 (cit. on pp. 20, 21, 23).

[10] II Clifton A. Ericson. *Hazard Analysis Techniques for System Safety*. John Wiley & Sons, Inc., 2005. Chap. Chapter 21: Hazard and Operability Analysis (cit. on pp. 12, 13, 17, 20–22).

[11] F. I. Khan. "Knowledge-based expert system framework to conduct offshore process HAZOP study". In: *2005 IEEE International Conference on Systems, Man and Cybernetics*. Vol. 3. 2005, 2274–2280 Vol. 3 (cit. on pp. 20–22).

[12] J.C. Parmar and F.P. Lees. "The propagation of faults in process plants: Hazard identification". In: *Reliability Engineering* 17.4 (1987), pp. 277 – 302. ISSN: 0143-8174. URL: http://www.sciencedirect.com/science/article/pii/014381748790093X (cit. on pp. 21, 22).

[13] M. Mansoori et al. "Application of HAZOP to the Design of Cyber Security Experiments". In: *2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA)*. 2016, pp. 790–799 (cit. on pp. 21, 23).

[14] International Standards Organization. *ISO 26262-1: Road Vehicles : Functional Safety - Part 1: Vocabulary*. Second. .Part 1: Vocabulary. ISO, 2018 (cit. on pp. 3, 15).

[15] P. Fenelon et al. "Towards Integrated Integrated Safety Analysis and Design". In: *ACM SIGAPP Applied Computing Review* 2 (Sept. 1998) (cit. on p. 92).

[16] Malcolm Wallace. "Modular Architectural Representation and Analysis of Fault Propagation and Transformation". In: *Electron. Notes Theor. Comput. Sci.* 141.3 (2005), pp. 53–71. URL: https://doi.org/10.1016/j.entcs.2005.02.051 (cit. on p. 92).