Building a safety case for a small sized product line of Fuel Level Display Systems

Antonio Gallucci
Master Student – Software Engineering

Internal supervisor
Barbara Gallina
Post-Doc Researcher
Mälardalens Högskola
barbara.gallina@mdh.se

External supervisor
Mattias Nyberg
Expert Engineer
Scania AB, Södertälje, Sweden
mattias.nyberg@scania.com

Examiner
Kristina Lundqvist
Professor
Mälardalens Högskola
kristina.lundqvist@mdh.se

Abstract:

ISO 26262 is an international standard valid for the automotive domain. It regulates all the activities to perform for developing safety critical systems in such domain. To be compliant with ISO 26262, all the required activities have to be performed and all the required work products have to be provided. Furthermore, in addition to develop a system in a safe way, following the safety standard guidelines, the achieved safety has also to be demonstrated. This is done through a safety case, a structured argument showing that a system is acceptably safe.

ISO 26262 focuses on single systems and does not contain guidelines for product lines. Product line engineering is a valid approach to systematize reuse, aimed at reducing the effort needed to develop similar systems. But, it loses its strength when dealing with safety critical systems, since it is not aligned with safety standards. Hence, when developing a safety critical product line in the automotive domain, the work products required by ISO 26262 have to be provided every time from scratch, including the safety case, for each single system of the product line.

This thesis work focuses on providing an approach for building and modeling a safety case for safety critical product lines in the automotive domain. Furthermore, the considered product line engineering approach is aligned with ISO 26262, through the inclusion of safety activities in the product line development process. Giving in this way, the concrete possibility to overtake to the current limitations, reducing the effort needed to develop and certificate each single system of a safety critical product line. To illustrate the validity of the proposed approach a safety critical product line developed by Scania is used as case study.

Acknowledgments

First of all, I would like to thank my internal supervisor, Barbara Gallina, for her precious suggestions and support during the entire thesis period. I could not find a better supervisor.

I would like to thank my external supervisor, Mattias Nyberg, for giving me the great opportunity to perform this thesis work within a wonderful company, such as Scania.

I would like to thank my examiner, Kristina Lundqvist for arousing in me a strong interest in the safety critical systems engineering field.

Special thanks to my friends, Achille, Niklas, Aleksandar, Rasul, Tolga, Daniel, Mahmoud and many more for the great time that we spent together during my stay in Västerås.

Last but not least, I would like to thank my family for their great support during all this wonderful experience, without which I never would have made it.

Table of Contents

Index of Figures

Index of Tables

Acronyms and Abbreviations

ASIL	Automotive Safety Integrity Level
BCS	Body chassi system
COO	Coordinator
E/E	Electrical and/or electronic
ECU	Electronic Control Unit
FLDPL	Fuel Level Display Product Line
EMS	Engine Management System
FLDS	Fuel Level Display System
FSR	Functional Safety Requirement
FTA	Fault Tree Analysis
GSN	Goal Structuring Notation
HAZOP	Hazard and Operability Analysis
ISO	International Organization for Standardization
PLE	Product Line Engineering
SysML	Systems Modeling Language
TSR	Technical Safety Requirement

1. Introduction

This introductive chapter is organized as follows: in section 1.1 the context and motivation of the thesis are explained; in Section 1.2 the contributions of this thesis work are presented; in Section 1.3 the document structure is described.

1.1 Context and motivation

ISO 26262 is a safety standard for the automotive domain. Quoting from the standard: "ISO 26262 is intended to be applied to safety-related systems that include one or more electrical and/or electronic (E/E) systems and that are installed in series production passenger cars with a maximum gross vehicle mass up to 3 500 kg." By now, trucks do not have to be compliant with the standard. However, it is likely that by 2016 they will have to [1].
ISO 26262 regulates all the phases of the entire lifecycle of the system, starting from the management and requirements specification phases up to the production release. The standard also defines the work products that have to be produced during the system lifecycle. A *safety case* (a structured argument used to show that the system is acceptably safe) is one of these work products required by the standard.
Arguments used to show that the process defined in the standard has been adopted, are called *process-based arguments*; instead, the ones used to show that the system behavior is acceptably safe are called *product-based arguments*. Process and product-based arguments are combined in a safety case.
As intended in ISO 26262, the safety case is aimed at showing the safety of one single system, and not of *product lines* that are families of systems having some common functionalities and other variable ones, which distinguish each single system.
Thanks to these common and variable functionalities, a systematic approach in reusing different artifacts resulting from all the phases of the system development process could lead to the easing of development of each single system. Such approach is called *product line engineering* (PLE) *approach*, and it could be extended considering also safety activities. Supporting in this way, the systematic reuse also for such activities, including the building of a safety case. Hence, easing not only the development, but also the certification[1] of the entire safety critical product line.

This thesis work has been performed in collaboration with Scania, that is one of the leading manufacturers of heavy trucks, buses, coaches and engines. To be ready by 2016, Scania is interested in continuing investigating ISO26262 as well as safety case provision. Thus this thesis builds on top of previous ones [2] [3] and it focuses on providing a systematic approach to build a safety case for a small-sized product line of Fuel Level Display Systems, which Scania considers being safety critical systems and need to be developed in compliance with the standard.

[1] In this thesis the term *certification* is used to denote demonstration of functional safety of a given product in compliance with a given standard.

1.2 Contribution

To build a safety case for a product line in compliance with ISO 26262, a product line development process is aligned with the safety standard, through the inclusion of the considered and required safety activities, in the product line development process. This makes possible the identification and management of commonalities and variabilities within the outcomes related to a safety activity in the development process, supporting in this way reusability, and reducing the effort to develop and certificate each single system of a product line. Furthermore, a product line modeling approach is provided for each considered activity, giving the possibility to represent commonalities and variabilities within the relative outcomes.

So far, product line development processes have not explicitly aligned with safety standards. Product line modeling approaches are available for different aspects of the product line development process, but they have not been explicitly selected and combined for a product line development process, in order to be compliant with ISO 26262. In this thesis work an alignment of the product line development process with the activities required by ISO 26262 is proposed. Several product line approaches are accurately selected and extended were needed, in order to highlight and manage commonalities and variabilities for each considered development activity and the relative outcomes, in the context of ISO 26262.

In addition to manage and model commonalities and variabilities, an approach to represent a safety case relating his common and variable parts to the commonalities and variabilities identified during all the considered activities of the product line development process is used. Giving a clear view, on how a careful identification and tracking of the common and variable aspects of a product line, impacts on the structure of the safety case. Giving in this way concrete reusing possibilities in order to reduce the time-to-market, development and certification effort for a safety critical product line. The validity of the presented approach is also shown, applying it to a real set of similar safety critical systems developed by Scania.

Finally, the paper *VROOM & cC: a Method to Build Safety Cases for ISO 26262-compliant Product Lines* [4] stemmed from this thesis work, and it has been accepted at the Next Generation of System Assurance Approaches for Safety-Critical Systems workshop (SASSUR).

1.3 Document structure

The rest of this thesis is organized as follows:

Chapter 2 is intended to present the background information needed to understand the topics discussed and analyzed in this thesis. This information is also used to build the proposed solution in Chapter 4 and 5;

Chapter 3 contains the problem formulation and its analysis. The main problem is presented in detail and it is divided into several sub-problems, each of them covering a different aspect, in order to simplify its resolution.

Chapter 4 explains how to use the available approaches to solve the identified sub-problems, describing step by step how to achieve the desired goal. Furthermore, available approaches limitations in relation to this thesis work are presented;

In Chapter 5 a general overview of the proposed alignment of the product line development process and the activities required by ISO 26262 is given. In addition the highlighted approaches limitations, within Chapter 4, are addressed.

Chapter 6 presents the case study describing how the proposed approach is applied on the Fuel Level Display Systems. All the considered activities required by ISO 26262 are performed, using the product line techniques described in Chapter 4 and the Chapter 5;

Chapter 7 concludes the thesis work by describing the benefits and limitations of this thesis work. Furthermore, directions for future work are presented.

2. Background

This chapter provides the necessary background information related to the topics covered in this thesis work. More specifically, In Section 2.1 safety critical systems are handled; In Section 2.2 ISO 26262 standard is described, with a focus on the sections that will be handled in this thesis; In Section 2.3 the concept of safety case is presented; In Section 2.4 safety case modeling is described; In Section 2.5 product lines and the product line engineering approach is described; In Section 2.6 feature diagrams are explained; In Section 2.7 activity diagrams are presented; In Section 2.8 SysML is described; In Section 2.9 the concept of Hazard analysis and some hazard analysis techniques are presented; Finally in Section 2.9 the different versions of the Fuel Level Display System are described.

2.1 Safety critical systems

Safety critical systems are those systems whose failure could result in loss of life, significant property damage, or damage to the environment [5]. Example of safety critical systems are the ones used in aircrafts, cars, trucks, solving important functions. *Failures* in these important functions (event that occurs when the delivered service deviates from the correct service [6]), can cause serious damages to the surrounding environment, e.g. auto-pilot function failure, in an aircraft, could cause loss of human life. Many examples of safety critical systems failures can be found in [7] [8] [9]. A failure is caused by an *error*, which is the deviation from the correct system behavior. While the adjudged or hypothesized cause of an error is called f*ault* [6].
This fault-error-failure sequence can leads to a hazard, which is an unsafe condition, which in turn can cause an harm. This chain of events is summarized in Figure 1.

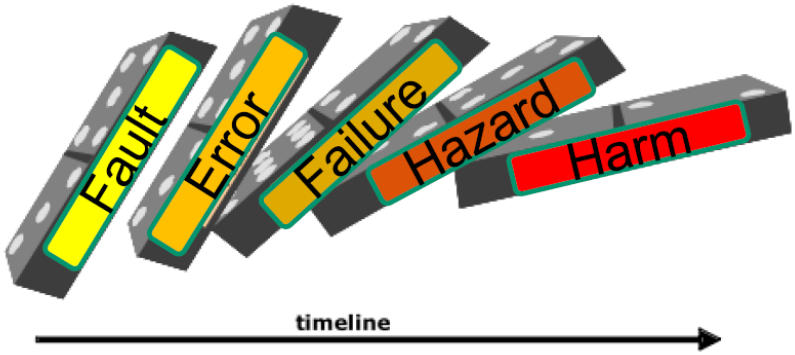

Figure 1:Chain of events leading to a harm taken from [10].

Since hazards related to safety critical systems can also cause severe injuries, e.g. loss of human life, the s*afety* must be ensured. Safety can be defined as the absence of catastrophic consequences on the user(s) and the environment [6], and it can be partially reached by performing adequate safety activities, e.g. hazard analysis (Section 2.8) and using appropriate safety measures, e.g. redundancy. The total safety of a system cannot be ensured, because there is always a residual risk that a failure can occurs. Also if the total safety cannot be reached the residual risk can be reduced below a minimum established threshold. Once the residual risk is acceptable, the system can be considered as acceptably safe, and can operate in the relative environment.

In order to ensure the acceptable system safety, the activities to develop safety critical systems could be dictated by safety standards. Since, the concept of safety is very general, there are safety standards that are more specific, which are written explicitly to manage sub-parts of the overall system safety, as for example the *functional safety*. The functional safety is the part of the overall safety which is related to the correct behavior of the system in terms of its inputs and outputs. An example of a functional safety standard is the ISO 26262, which is valid for the automotive domain.

2.2 ISO 26262

ISO 26262 [11] is an international safety standard for the automotive domain. Nowadays it is not mandatory for companies that build E/E systems for trucks and couches, but likely it will be required from 2016 [1]. The international standard covers all phases of the system development process, starting from the management to the maintenance and decommissioning phases. It is composed of 10 sections, organized in clauses, covering different aspects of the management and development processes.

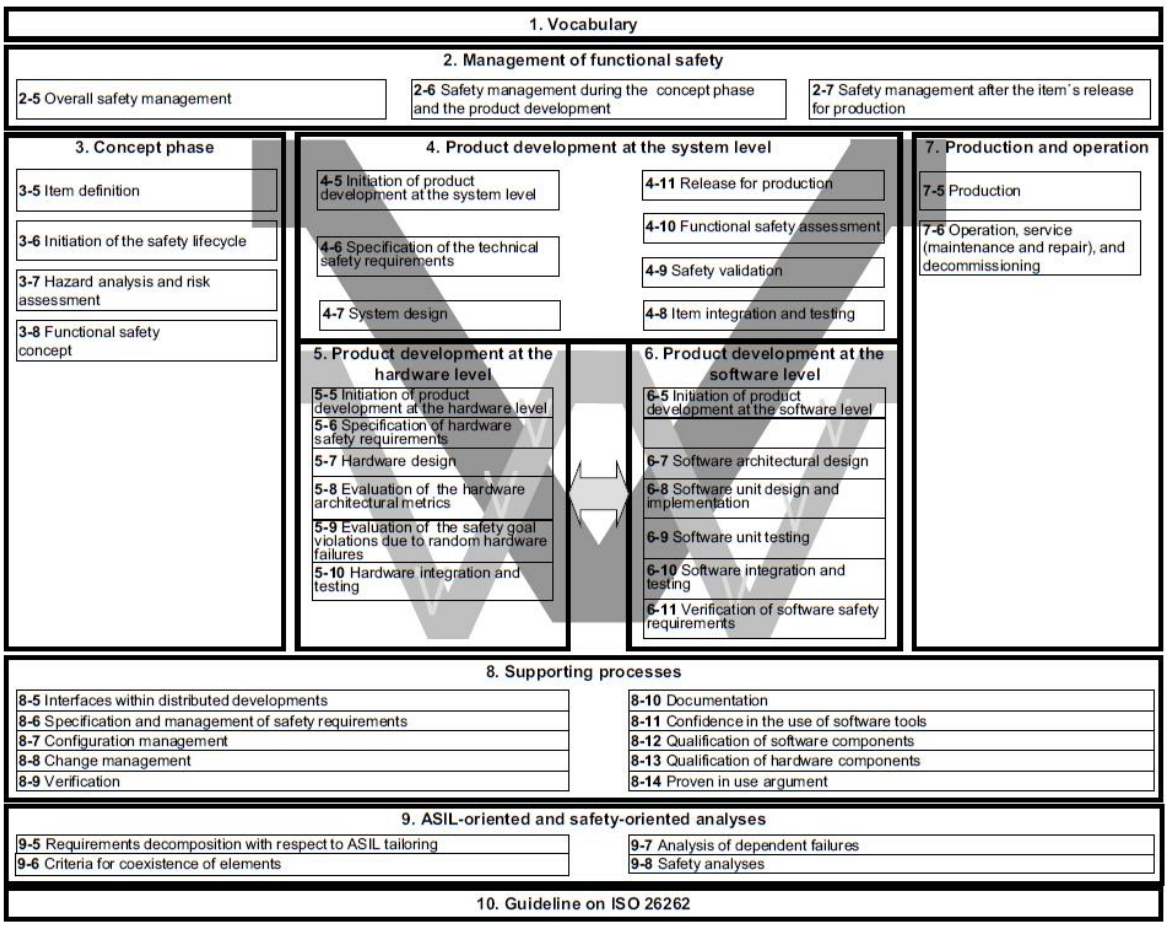

Figure 2: Overview of ISO 26262 taken from [11]

In Figure 2 the sections composing the standard and how they are organized are presented. ISO 26262 is based upon a V-model and it is organized as follows: in Section 1 the terms, definitions and abbreviated terms for application in all parts of ISO 26262 are specified; in Section 2 the requirements for functional safety management are presented; in Section 3 the requirements for the performing of the concept phase for automotive application are specified, e.g. requirements on the definition of functional and safety requirements and performing of hazard analysis; In Section 4, 5 and 6 the requirements for product development respectively at the system, hardware and software level are presented; In Section 7 the requirements for production, operation, service and decommissioning are described; In Section 8 the requirements for supporting processes, including overall management of safety requirements, documentation, verification and confidence in the use of software tools are presented; In section 9 the requirements for Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses are described; Finally, in Section 10 general guidelines on ISO26262 are specified.

Since ISO26262 has a very wide scope, this thesis work focus mainly on the third section and part of the fourth one. In the next chapters, Section 3 and 4 of ISO26262 are described more in detail.

2.2.1 ISO 26262-3 Concept phase

The Concept phase section of the standard is composed by four clauses, which are recalled in the following list: Item definition, Initiation of the safety lifecycle, Hazard analysis and risk assessment and Functional safety concept. These four clauses are recalled below.

Item definition

The goal of the item definition clause is to define and describe the item, its dependences on, and interaction with, the environment and other items [11]. Hence, functional and non-functional requirements, boundary, interfaces and assumptions regarding interaction with other systems need to be provided. The result of this phase is the Item Definition work product, and it is a prerequisite for the next clauses.

Initiation of the safety lifecycle

The Initiation of the safety lifecycle clause defines the safety life cycle activities to perform, distinguishing whether it is a new system or modification of an existing one. In the second case, an evaluation of the impact of the changes on the item and its environment need to be performed and documented in the Impact analysis work product. In addition if safety activities need to be tailored or if work products are missing or not compliant with the standard, the necessary activities need to be specified and documented in the safety plan .

Hazard analysis and risk assessment

The Hazard analysis and risk assessment clause defines the activities to carry out in order to identify and categorize the hazards related to the system and its behavior. Furthermore, the clause describes requirements on the activities to be performed for defining the necessary safety goals in order to prevent or sufficiently mitigate the identified hazards, avoiding in this way unreasonable risks.

According to ISO 26262, after that the hazards have been identified they have to be classified. There are different ways to classify them, but the ISO 26262 explicitly defines one method combining *severity, controllability* and *exposure*.

The severity is intended as the entity of the damage that the system could lead to in that hazardous situation. ISO26262 defines four levels of severity going from S0 to S3, where S3 is the highest one. Each level of severity is described in more detail in Table 1.

	Class			
	S0	**S1**	**S2**	**S3**
Description	No injuries	Light and moderate injuries	Severe and life-threatening injuries (survival probable)	Life-threatening injuries (survival uncertain), fatal injuries

Table 1: Classes of severity taken from [11]

Instead, the controllability is the measure of the capacity to bring back the system in a safe state or to manage the hazardous situations. As for the severity, also the controllability has four different levels, going from C0 to C3, where C3 is the highest one. Each level of controllability is described in more detail in Table 2.

	Class			
	C0	C1	C2	C3
Description	Controllable in general	Simply controllable	Normally controllable	Difficult to control or uncontrollable

Table 2: Classes of controllability taken from [11]

Finally the exposure is the probability that the event can occur, in the specified operational situation. ISO 26262 identifies 5 levels of exposure going from E0 to E4, where E4 is the higher one. Each level of severity is described in more detail in Table 3.

	Class				
	E0	E1	E2	E3	E4
Description	Incredible	Very low probability	Low probability	Medium probability	High probability

Table 3: Classes of Exposure taken from [11]

Based on these parameters an *Automotive Safety Integrity Level* (ASIL) is assigned to the hazard, assuming a value between A and D, where A is the lowest and D the higher integrity level. The ASIL can be considered as a measure of the rigor needed to mitigate the hazards, imposing requirements on the processes to follow during the system development process. In addition to these ASILs, there is another level, i.e. QM, indicating that there is no need to specify additional requirements. The combination of severity exposure and controllability and the relative ASIL are described in Table 4.

Severity class	Probability class	Controllability class		
		C1	C2	C3
S1	E1	QM	QM	QM
	E2	QM	QM	QM
	E3	QM	QM	A
	E4	QM	A	B
S2	E1	QM	QM	QM
	E2	QM	QM	A
	E3	QM	A	B
	E4	A	B	C
S3	E1	QM	QM	A
	E2	QM	A	B
	E3	A	B	C
	E4	B	C	D

Table 4: ASIL determination taken from [11]

For example having S3 as severity level, E3 as exposure level and C2 as controllability level the relative ASIL will be C.

According to ISO 26262, the successive step, after identifying and classifying hazards, is to define safety goals. *Safety goals* are requirements defined to eliminate or sufficiently mitigate

one or more hazards, and they are used to translate hazards and their ASIL to the relative mitigation measures. An ASIL is assigned to each identified safety goal, and this is inherited from the correspondent hazard. If a safety goal corresponds to more than one hazard the highest ASIL is considered. Furthermore, this clause requires that the hazard analysis, risk assessment and the safety goals shall be verified in accordance with the standard.

The result of this clause is documented in three work products:

1. Hazard analysis and risk assessment
2. Safety goals
3. Verification review report of the hazard analysis and risk assessment and the safety goals work products.

Functional safety concept

The goal of this clause is to decompose the safety goals into *Functional Safety Requirements* (FSRs) and allocate them to elements of the system architecture.
FSRs specify the safety measures to be applied to the system architecture in order to meet the relative safety goal, and for each safety goal at least one functional safety requirement have to be specified.
In addition, the functional safety requirements need to be verified, showing that are consistent and compliant with the safety goals and mitigate or avoid the hazardous events.

The result of this clause is documented in following work products:

1. Functional safety concept
2. Verification report of the functional safety concept

2.2.2 ISO 26262-4 Product development at system level

The goal of the Product development at system level section is to define requirements concerning the activities to perform at system level in the development process. It is composed by five clauses: Initiation of product development at the system level, Specification of technical safety requirements, System design, Item integration and testing and Safety validation.

In this section only the Specification of technical safety requirements and System design clauses are described, since the other ones are out of the scope of this thesis.

Specification of technical safety requirements

The goal of the Specification of technical safety requirements clause is to specify requirements on the decomposition of FSRs, in *Technical Safety Requirements* (TSRs) and their allocation.
TSRs shall decompose and refine FSRs, and for each FSR at least one TSR has to be specified. TSRs shall detail the FSRs that are specified at item level into safety requirements that are specified at system level. TSRs shall specify the necessary safety mechanisms and measures to implement FSRs.
Furthermore, TSRs shall also be verified to provide compliance with FSRs and the preliminary architectural design assumptions.

The work products resulting from this clause are:

1. Technical safety requirements specification
2. System verification report
3. Validation plan

System design

According to ISO 26262 the next step after the definition of TSRs, is to define a system design that meets FSRs and TSRs. This is the goal of the System design clause.
A second goal of this clause is also to verify that the system design effectively comply with FSRs and TSRs.
Furthermore, in order to avoid systematic failures, through the identification of their causes, safety analysis on system design shall be performed. Deductive and inductive analysis (Section 2.8) can be performed. For systems with ASIL C or D both type of analysis are mandatory, while for systems with ASIL A or B only inductive analysis is mandatory. This information is summarized in Table 5.

Methods		ASIL			
		A	B	C	D
1	Deductive analysis[a]	o	+	++	++
2	Inductive analysis[b]	++	++	++	++
[a] Deductive analysis methods include FTA, reliability block diagrams, Ishikawa diagram.					
[b] Inductive analysis methods include FMEA, ETA, Markov modelling.					

Table 5: System Design Analysis taken from [11]

The work products resulting from this clause are:

1. Technical safety concept
2. System design specification
3. Hardware-software interface specification (HSI)
4. Specification of requirements for production, operation, service and decommissioning
5. System verification report (refined)
6. Safety analysis reports

2.3 Safety case

When a safety critical system is developed, in order to be certified there is need to show that it is acceptably safe. This can be done through the use of a safety case. More specifically a safety case should communicate a clear, comprehensive and defensible argument that a system is acceptably safe to operate in a particular context [12].

A safety case is composed by three main elements: Requirements, Arguments and Evidences. Requirements describe the objectives to obtain, while Arguments can be used to show the relation between different requirements. Instead, Evidences are used to support requirements.
Between requirements and evidences there is a strict relation. If a requirement is not supported by evidence is unfounded, while if an evidence is used without referring to any requirement is unexplained [12]. Hence, fallacies in the safety case might undermine the safety case itself, as described in [6] and more generally in [13].

The type of evidence required by a safety case depends on the certification method. As described in [14] there are basically two general types of certification approaches, which determine the type of evidence used in the certification process:

1. Prescriptive
2. Performance-based or goal-setting

Prescriptive approaches require evidences showing that the system behaves in a safe way or that the process defined in the guidelines or standard have been followed. While, Performance-based or goal-setting approaches focus on desired, measurable outcomes, rather than system behavior or processes.
A prescriptive approach will be considered for this thesis.

The evidences used to show that the processes defined in the safety standard have been adopted are called process-based evidences, while the ones used to show that the system behaves in a safe way and its usage does not lead to unacceptable risks, are called product-based evidences. As discussed in [15] product-based and process-based evidences are essential elements of a safety case.

Hence, the safety case should be a structured argument showing that all the processes defined in the standard have been performed, through process-based arguments, and also how safety goals have been met, and all identified hazards have been sufficiently mitigated with the appropriate level of integrity, through product-based arguments, being in this way a valid argument for the system safety.

Similarly to the system development lifecycle adopted in ISO 26262 (V-model), the safety case has its own development lifecycle (preliminary, interim, operational status) as shown in Figure 3.

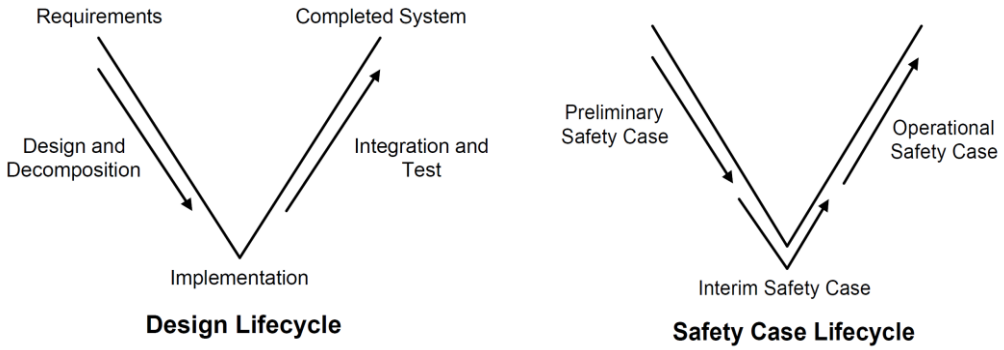

Figure 3: Design life-cycle and Safety case life-cycle taken from [12]

The Preliminary safety case is the first version of the safety case and it is built after the definition and review of the system requirements specification. Then, after the initial system design and preliminary validation activities it is refined, and the refined version is called Interim safety case. The last version of the safety case is the Operational safety case and it is finalized just prior to in-service use, including complete evidence of satisfaction of systems requirements [12].

Hence, the building of the safety case should not be an activity performed at the end of the system lifecycle but it should be considered during all system development life-cycle. An example of how a preliminary safety cases is managed during the design phase is presented in [16].

In the context of ISO 26262 the safety case is a mandatory work product and as such, it is needed to certify systems in the automotive domain. According to the safety standard the safety case should progressively compile the 'work products' that are generated during the safety life-cycle [11]. Hence, it could be interpreted as a simply demonstration that the required work products exists. Instead, to be a valid argument for system safety, it should contain both product-based and process-based arguments as discussed in [17].

2.4 Safety case modeling

To represent a safety case a textual or a graphical-based notation can be used. Examples of textual based notations are: normal prose, structured prose, argument outline, mathematical proof or LISP style as described in [18]. Instead, example of graphical notations are the *Claim Argument Evidence* (CAE) notation [19], and the *Goal Structuring Notation* (GSN) [20] [21]. In Section 2.4.1 a general overview of GSN is given to the reader.

2.4.1 Goal Structuring Notation

GSN is a graphical notation to represent safety cases. An overview of the main elements of GSN is given in Figure 4.

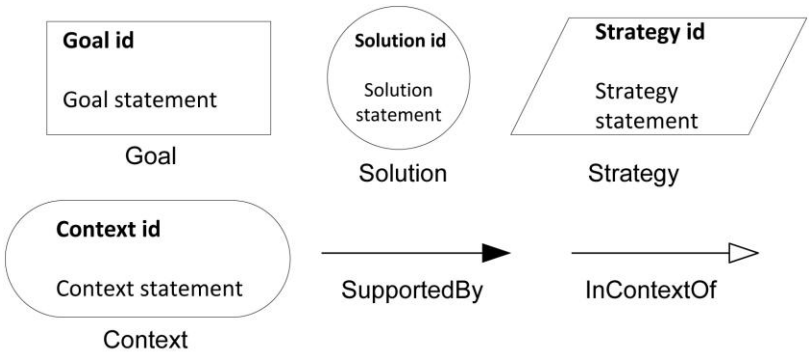

Figure 4: Principal elements of the Goal Structuring Notation

As Figure 4 shows, the main GSN elements are: *Goal, Solution, Strategy, Context*, S*upported by* relation and *In context of* relation. The description of the elements follows:
A G*oal* is a statement about the system safety that needs to be supported directly or indirectly by solutions;
A *Strategy* can be used to describe the relation between a goals and its sub goals, representing the argument used to decompose the main goal;
The *Context* element instead, is used to describe the circumstances in which the system safety is argued;

SupportedBy relation allows inferential or evidential relationships to be documented. Inferential relationships declare that there is an inference between goals in the argument. Evidential relationships declare the link between a goal and the evidence used to substantiate it;
InContextOf relation declares a contextual relationship [21].

Combined together these elements define a goal structure. An example of goal structure is given in Figure 5.

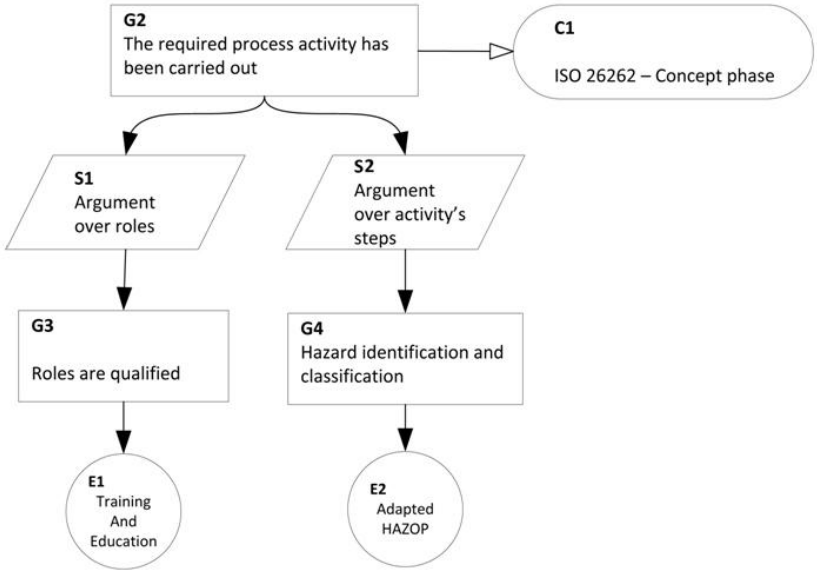

Figure 5: An Example Goal Structure taken from [3]

In the presented goal structure, the main goal G2, states that the required process activity has been carried out, in the context of the concept phase of ISO 26262. In this case, G2 is clearly related to process-based aspects, and it is broken down in two sub-goals through two different strategies: one over the roles (S1) and the other over the activity's steps (S2). S1 is supported by the goal G3, stating that the roles are qualified, and this goal is supported directly by the evidence E1. Instead S2 is supported by the goal G4, stating that the hazards have been identified and classified, and it is directly supported by the evidence E2.

In order to support safety case modularity, GSN has been extended [21] with modular extensions. Giving in this way the possibility to construct compositional safety cases as shown in [22]. The extended GSN symbols are represented in Figure 6.

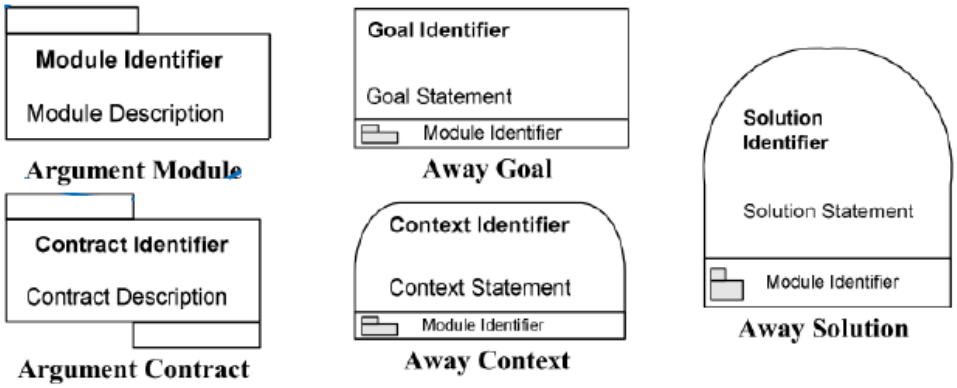

Figure 6: Elements of the GSN extended taken from [21]

A *Module* contains a part of the goal structure. An A*way goal* instead, is a goal used to support a claim within a module, but defined in another module. *Away solution* and *Away context* are defined in the same way. Finally, a *Contract* is used to relate two modules, and it defines how a claim in a module is supported by a goal in another module.

There can be some cases where a successful goal structures can be reused and applied also to a different context. In that case in order to systematize goal structures reuse a pattern can be defined [23] [24]. *Patterns* are defined to systematically reuse the same successful goal structure in different contexts.
In order to define patterns GSN has been extended [21] with the elements represented in Figure 7.

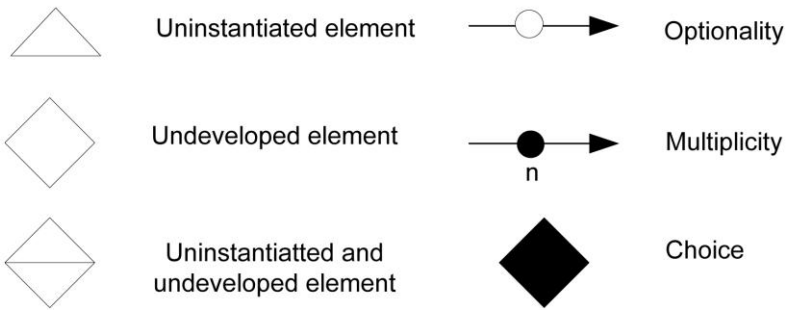

Figure 7: Pattern modeling elements

In a GSN pattern a goal could refer to a generic element, e.g. System X, instead of a specific systems, in order to be more general to allow the utilization of that goal structure for different systems. When a GSN element, e.g. a goal, contains a variable aspect, a white triangle at the base of the relative GSN symbol is used. In the case of a goal, when the pattern is used, the goal has to be instantiated considering the actual value the variable aspect.
Instead, when a goal needs to be further developed, a diamond shape at the bottom of the goal symbol has to be used.
A goal can be both un-instantiated and undeveloped and this can be represented with a

diamond shape, with a horizontal line in the middle.
There can be some cases where a part of the goal structure is optional and can or cannot be present in the entire goal structure, in this case the optionality is represented through a white circle on the relative relation. Hence, when the pattern will be instantiated the optional part could be present or not based on the actual system for which the pattern is used.
Furthermore, an element could also be present multiple times in the goal structure with little variations. This multiplicity can be represented by a black circle on the relative relation and by an *n* indicating the number of possible instances.
Finally there can be the case where an option has to be taken among different choices and this can be represented by a diamond as starting point for different relations, each of them representing a choice.
A simple example of GSN pattern is shown in Figure 8.

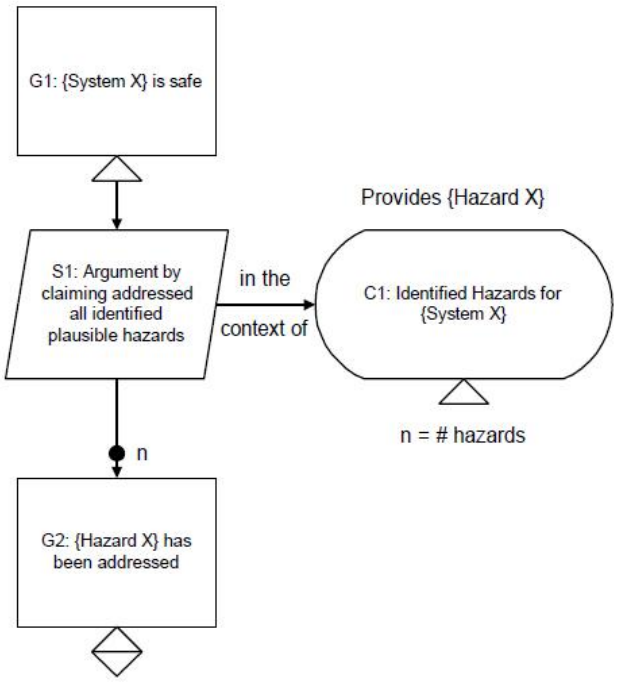

Figure 8: Hazard avoidance pattern taken from [25]

As Figure 8 shows, the main goal of the pattern claims the safety of System X, where System X denotes a variable that needs to be instantiated with the name of the actual system. This goal is supported by claiming (by arguing) that all the plausible identified hazards have been addressed, in the context of the identified hazards for the System X. The number of hazards is represented by the variable *n* and for each hazard there is a different goal claiming that it has been addressed. This goals needs to be further developed. Hence, there will be n goals of the same type of the G2 goal and each of them will be instantiated with the identification name of the hazard and developed in a different way. Another patter can be combined with this, to show how each hazard has been addressed. Several GSN patterns are available in [25].

Using the available GSN modular and pattern extensions an approach for building configurable safety has been proposed in [26]. To support this approach the *Obligation* symbol has been introduced. An obligation describes a condition related to a point where a choice has to be made, within the goal structure. There can be two types of conditions:

Intrinsic and *extrinsic*. Intrinsic conditions are the ones related to other elements of the same goal structure, e.g. the presence of a goal can imply the presence of another one. Instead, extrinsic conditions are the ones related to external aspects of the safety case. E.g. a system design choice can influence the goal structure. An example of goal structure including an obligation is shown in Figure 9.

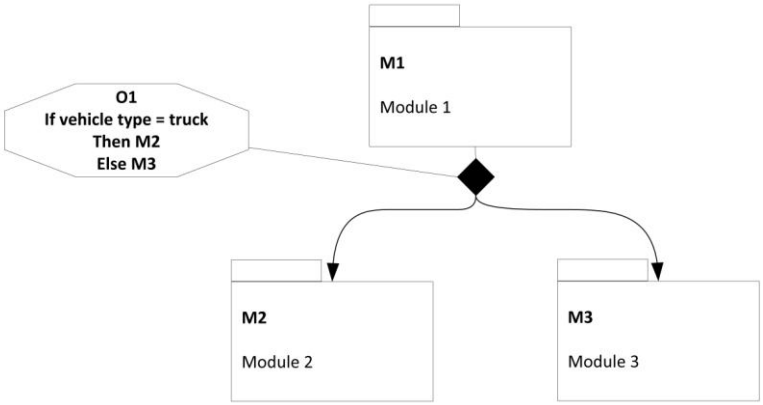

Figure 9: Example of configurable goal structure

As Figure 9 shows, in the presented goal structure there are tree modules: M1 is mandatory, while M2 and M3 are related by a XOR alternative. Thus either Module 2 or Module 3 will be present in the instantiated goal structure, depending on the vehicle type, as specified in the obligation O1.

2.5 Product line engineering

A (Software) product line is a set of (software-intensive) systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way [27].
A *product line engineering* (PLE) process [28] [29] is based on the identification and organization of common and variable aspects among the systems composing the product line, and using them to define artifacts, e.g. requirements or system architecture, for the entire product line, with the possibility to adapt them to each single system that compose it. Common aspects are called *commonalities* and variable aspects are called *variabilities*.
A variability is represented by a *variation point* and the different possible choices for a variation point are called variants. Configuring all the variation points of product line it is possible to define a *product line member* (system).

The product line engineering process (Figure 10) is divided basically into two phases: *domain engineering and application engineering* phases.
During the domain engineering phase the entire product-line is defined, based on common and variable aspects identified during variability and commonality analysis [30]. During the application engineering phase instead, single products are defined configuring the variation points defined during the domain engineering phase.

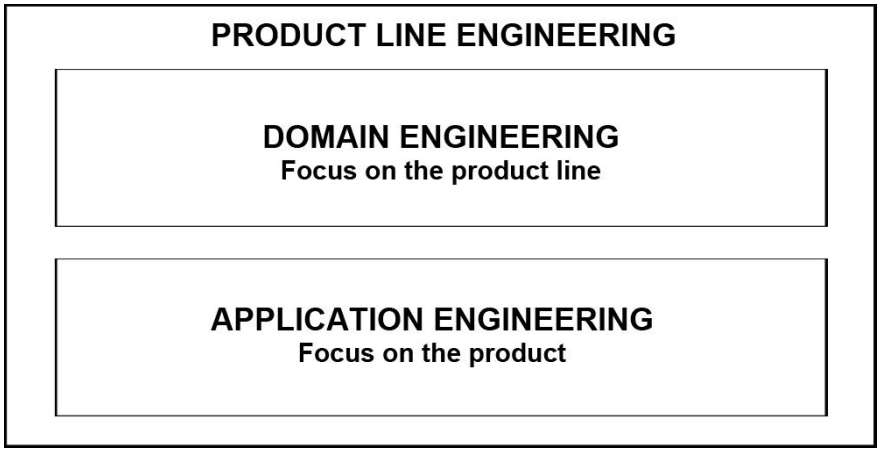

Figure 10: Product line engineering phases

This engineering approach highly supports the systematic reuse, contributing in saving time and costs to develop new similar systems. The effort required during the domain engineering phase will be significant, but in the long run the time needed to create new systems or update existing ones will be significantly decreased.

Product lines can also be developed in the context of safety critical systems (Section 2.1). In that case they are called safety critical product lines, and as such there is need to perform all the safety activities necessary for safety critical systems.

2.6 Feature diagrams

Feature diagrams is a means to represent commonalities and variabilities for a product line in terms of *features*, where a feature describes product characteristics from various stakeholder views: end-user, customer, analyst, architect, developer, etc [31]. Furthermore during the domain engineering phase they are an effective way to identify variability among different products in a domain [32].

In addition to represent common and variable features of a product line, with feature diagrams it is also possible to represent the relations among features. There can be three types or relations: *mandatory*, *optional* and *alternative*.
The mandatory relation indicates that the relative feature will be always present in all the product line configurations. Being in this way, core features of the product line.
The optional relation indicates that the relative feature can be optional. This means that a feature can be present in a system of the product line, but absent in another.
Finally the alternative relation is related to more than one feature. It indicates that only one of the relative features can be chosen. An example of feature diagram can be seen in Figure 11.

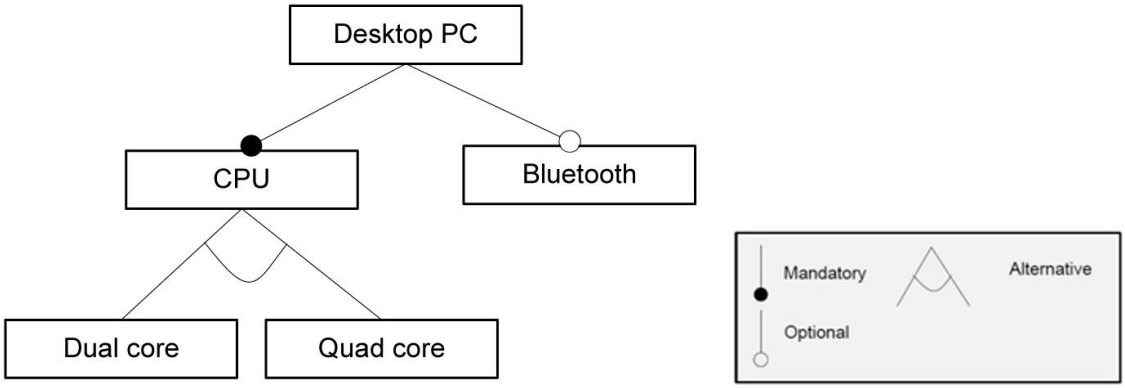

Figure 11: Feature diagram example

This diagram should be interpreted as follows: Desktop PC has a CPU which is mandatory, meaning that a Desktop PC will always have a CPU. Bluetooth functionality, instead is optional, meaning that the user that by the PC can choose to have this functionality or not. Furthermore, for the same PC it is possible to have a dual core CPU or a quad core one, but not both of them.

As we saw in this example, feature diagrams can be used for the representation of configurable aspects in a product line, hence defining allowed product line configurations through different types of relation.
In addition to the intra-constraints on the feature selection within the same feature diagram, there could be also external constraints, influencing the feature selection. E.g. there could be constraints influencing directly the selection or exclusion of some features. These constraints could derive for example from environmental, human or technical choices, influencing in this way the product line configuration.
In the context of the example shown in Figure 11, if, for example, the context in which the PC will be used is an elementary school a Dual core CPU could be required. Because the software used in that level of education does not require powerful CPUs. While, for Universities a Quad core CPU could be used for the PCs, because university students could require powerful CPU to run their software.

2.6.1 Usage context

An important factor influencing the feature selection is the *usage context*. Informally, the usage contexts are any contextual settings in which a product is deployed or used, which can be detailed in terms of user, physical, social, business, operating environments, etc [33].
The usage context can also be represented through feature diagrams. In Figure 12 is presented the usage context in relation to the previous example, with the relative constraints on the feature selection of the feature diagram shown in Figure 10.

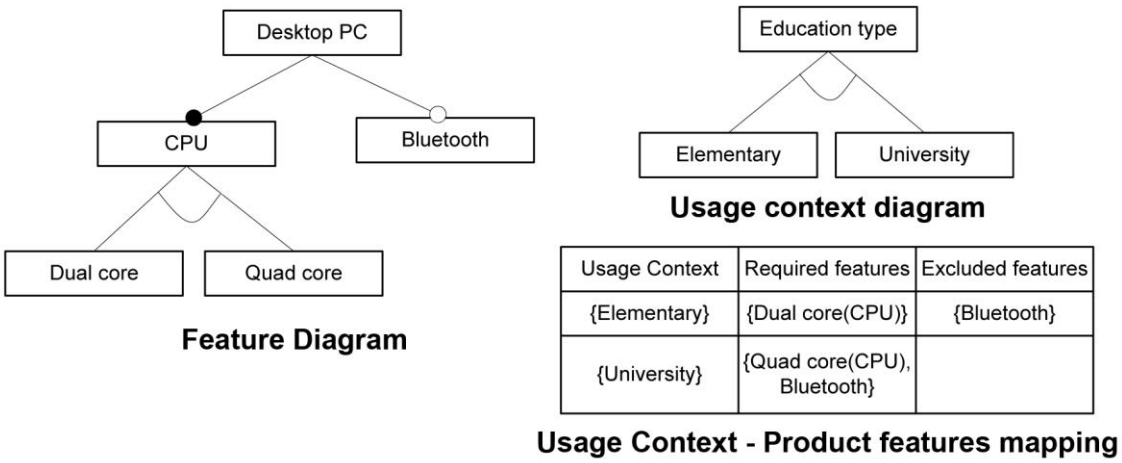

Figure 12: Usage context example

As we can see from Figure 12, a choice on the usage context have a direct impact on the feature selection of the feature diagram of the Desktop PC. Using a mapping table it is possible to define relations between the Usage context and the Desktop PC feature diagram. For example, if the type of education is Elementary then a required feature for the PC is the dual core CPU, while an excluded feature is the Bluetooth. Instead, if the type of education is University then, a required feature is the Quad core CPU and the Bluetooth.

2.7 Object Management Group (OMG) Systems Modeling Language (SysML)

The *Object Management Group's SysML* (OMG SysML) is a general-purpose graphical modeling language for representing systems that may include combinations of hardware, software, data, people, facilities and natural objects. [34]
OMG SysML include diagrams that can be used to specify system requirements, behavior, structure and parametric relationships [35].
The focus of this chapter is on the behavior and structure aspects of OMG SysML. In more detail, the focus is on block definition and internal block diagrams, and activity diagrams.

2.7.1 Activity diagrams

An *activity diagram* represents the flow of data and control between activities [35]. The basic elements of activity diagrams are represented in Figure 13.

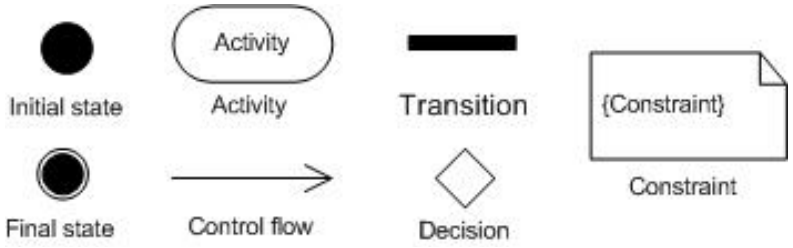

Figure 13: Basic element of an activity diagram

The start and the end of the activities is represented respectively from the *Initial state* and *Final state* symbols. *Activities* are represented through rounded rectangles. The arrows represent the *control flow* (order in which the actions happen). Diamonds represent *decisions*. When one arrow enters in the diamond and two or more exit, it is called *decision node*. While, if two or more arrows enter and one exits from the diamond, it is called *merge node*. Bars instead, represent the start and the end of *concurrent activities*. The constraint symbol instead is used to represent *constraints*. When it is used in combination with a decision node, then there is the so called *parameterization* of the decision nodes.
An example of activity diagram is shown in Figure 14

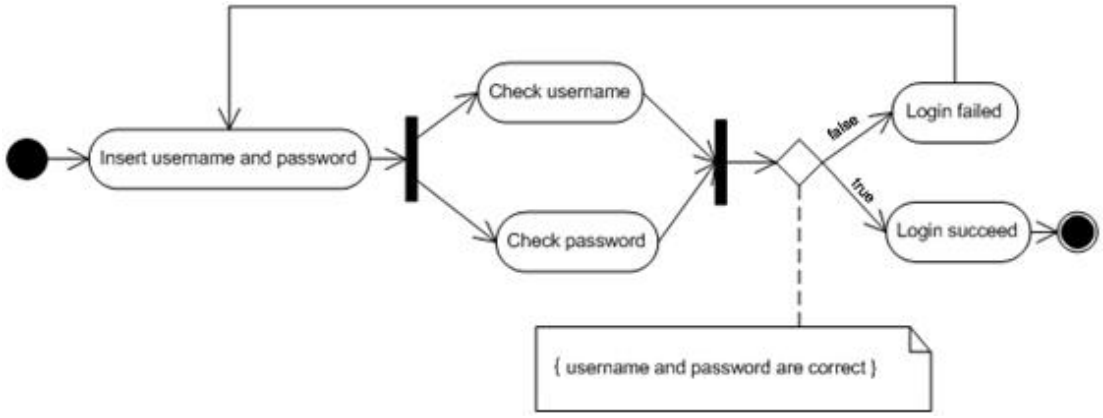

Figure 14: Example of an activity diagram

As Figure 14 shows, the first activity is inserting of username and password. After that there are two parallel activities, one checking the validity of the username and another one checking the validity of the password. Then, if the username and password are correct the login succeed and the activities end, while if they are not correct another possibility to insert them again is given, and this activities flow is repeated until that the inserted username and password are correct. This method of representing variability in activity diagrams has been taken from [36].

2.7.2 Block definition and internal block diagram

The system structure in OMG SysML is represented by *block definition diagrams* and *internal block diagram*. A block definition diagram describes the system hierarchy and system/component classification. The internal block diagram describes the internal structure of a system in terms of its parts, ports, and connectors [35].
An example of block definition and internal block diagrams is shown in Figure 15.

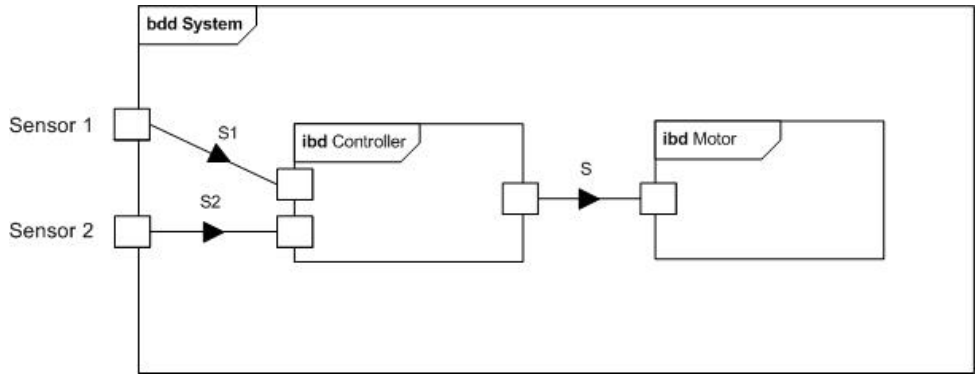

Figure 15: Example of block definition and internal block diagrams taken from [37]

In Figure 15 a system is represented, through a block diagram, with two components, i.e. Controller and Motor, through internal block diagrams. The interfaces are represented through ports and the relations through links. From Figure 15 we can see that the system is receiving input from two sensors and the values are read by the Controller, and after that the information is elaborated and sent to the Motor component.

2.7.3 Block definition and internal block diagram for product lines

In EAST-ADL [38] the block definition and internal block diagram has been used also to model product lines. In Figure 16 an example of how these diagrams are used to represent variability for product lines is shown.

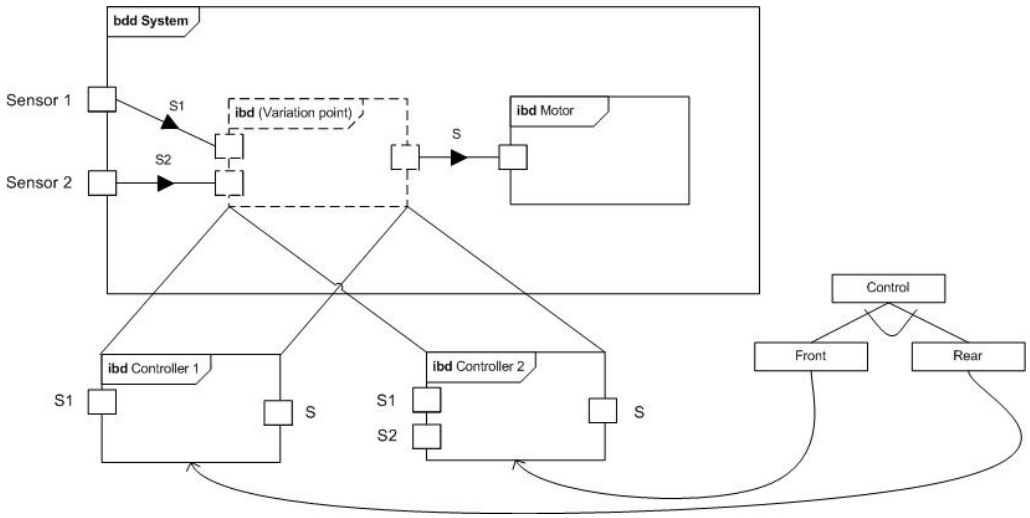

Figure 16: Example of variability with block diagrams taken from [37]

As Figure 16 shows, a variation point is indicated with an element with a dotted border. Then for the given variation point each variant is specified. We can also see that when the variant with the Controller1 is selected the S2 port is not present in the product line configuration. Furthermore, there is a mapping between each variant and the relative feature through a configuration link. This is to be able to define which variant for a given variation point, in the product line design model, has to be selected based on the correspond feature in the feature diagram. In this way it can be possible to define the impact of a feature selection on the

product line design.

In simple cases instead of defining a variation point, it could be enough to specify a variation group containing all the variants as shown in Figure 17.

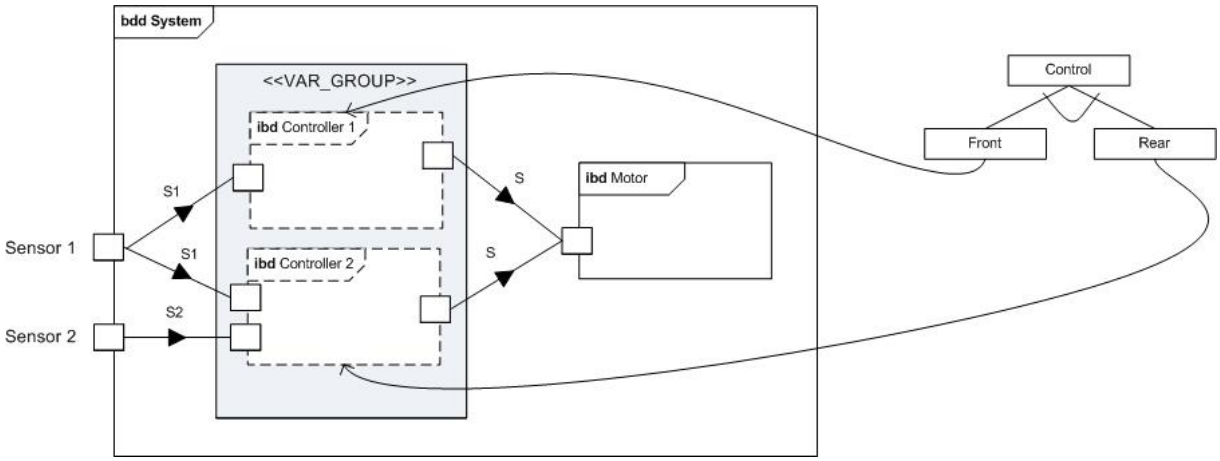

Figure 17: Example of variability with block diagrams taken from [37]

As Figure 17 shows, this time there is not a variation point, but all the variants are represented in the *variation group* <<VAR_GROUP>>. The achieved result is the same as the example in Figure 16.

2.8 Hazard Analysis

The *Hazard analysis* [39] is a technique used to avoid failures in a system through the identification and analysis of hazards. Hazards are situations that could lead to a harm, causing also severe injuries to human life and damages to the surrounding environment.
The hazard analysis can be performed at different levels. The *Preliminary Hazard Analysis* (PHA) is performed at the first stages of the system before the design phase; the *System Hazard Analysis* (SHA) is performed instead during the system design phase; and the *Sub-System Hazard Analysis* is performed at sub-system level.
There can be some overlaps between hazard analyses. In this case the common hazards have to be merged together. Usually the hazard analysis at a lower level (e.g. sub-system level) is a refinement of the one of higher level (e.g. system level), since there is more detailed information available.

In general there are two types of hazard analysis approaches: inductive and deductive approaches. With inductive approaches a failure is observed and the relative consequences are evaluated (*forward search*). Instead, with deductive approaches failures are analyzed and the relative causes are deducted (*backward search*). Furthermore, there are also hazard analysis techniques combining both inductive and deductive approaches. These techniques are called *bowtie* techniques, since they combine forward and backward searching techniques.
An available techniques for the inductive hazard analysis is the *Failure mode, Effects and Criticality Analysis* (FMECA) [39]. An example of deductive hazard analysis is the Fault Tree Analysis (FTA) (Section 2.9.2). Instead, an example of hybrid hazard analysis is the HAZard

and OPerability analysis (HAZOP) (Section 2.9.1).

2.8.1 Hazard and operability analysis

Hazard and Operability analysis (HAZOP) [40] is an hybrid hazard analysis technique, to analyse system deviations and determine their effects. HAZOP analysis is based on the analysis of deviations of the normal behaviour of the system. Identification of such deviation is done systematically using a set of attributes (e.g. value, occurrence, timing) combined with guide words (e.g. higher, lower, omission, commission, early, late).
By combining attributes and guide words it is possible to analyse system deviations and evaluate their effect, discovering in this way potential hazards. Not all the possible combinations are considered, but only the ones that have a plausible meaning. For example a plausible combination of attributes and guide words could be, value higher, value lower, occurrence omission, occurrence commission etc.
These deviations are analysed for an item of the system, e.g. system function, and the relative consequences are derived. The result of this analysis is stored in the HAZOP table. The basic structure of a HAZOP table is represented in Table 6.

ID	Item/interaction	Attribute	Guide word	Deviation	Consequences	Causes

Table 6: Basic structure of a HAZOP table

The information included in the HAZOP table presented in Table 6 are:

- ID: The identification of the deviation;
- Item/interaction: The item that is being considered, e.g. flow of signals, messages, data, system function etc.;
- Attribute: Attribute being considered for the deviation, e.g. value, occurrence, timing etc.;
- Guide Word: A guide word that has a plausible meaning combined with the attribute, e.g. higher, lower, commission, omission etc.;
- Deviation: The deviation from the correct system behaviour derived from the relative combination of the attribute and guide word.
- Consequences: The consequences of the considered deviation;
- Causes: The possible causes of the considered deviation;

Using this technique many hazards of a system can be discovered. Identified hazards can be used to improve system safety, through the implementation of safety measures addressing the identified hazards. This process of hazard mitigation is called risk assessment process.

In order to be compliant with ISO 26262, HAZOP table has been extended in [10], adding four additional columns, i.e. Exposure (E), Controllability (C), Severity (S) and ASIL. The extended table is presented in Table 7.

ID	Item/interaction	Attribute	Guide word	Deviation	Consequences	**E**	**C**	**S**	**ASIL**	Causes

Table 7: Extended HAZOP table taken from [10]

These additional columns corresponds to the parameters required by ISO 26262 described in Section 2.2.1 for hazard analysis. In this way, for each identified hazard it is possible to define an ASIL, making the HAZOP analysis compliant with the safety standard.

2.8.2 Fault tree analysis

The *Fault Tree Analysis* (FTA) [39] is a deductive hazard analysis technique carried out at the system level. It is performed using a top-down approach, starting from one hazard and analyzing which and how events contribute to the hazardous event.
The result of the FTA is the *fault tree* . The fault tree is a tree where the top node is the hazard being considered, named *top event*. It can be connected to his children through logic gates. There are several types of logic gates. The main ones are the *AND* and *OR gates*.
When events are connected to their parent through an AND gate, all of them need to occur in order to make the parent event occurring. While, if they are connected through an OR gate, only one of them needs to occur in order to make the parent event occurring. The events at the base of the tree are named *basic events*.
Events in the fault tree are represented through rectangle shapes and basic events through circle shapes. Instead, the *AND* and *OR* gates are represented by the symbols in Figure 18.

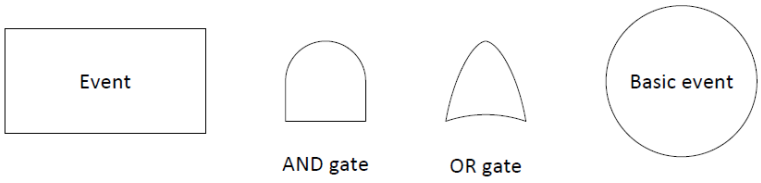

Figure 18: Fault tree symbols

A simple example of fault tree is presented in Figure 19.

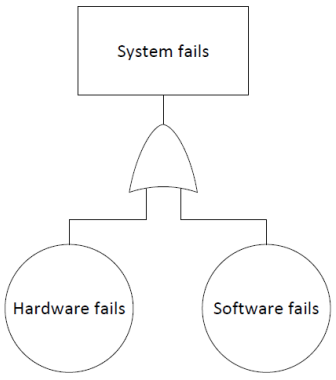

Figure 19: Example of Fault tree

As Figure 19 shows, the system can fail if the hardware or the software fails. The basic events are connected to the top event through an OR gate. This means that if there is a hardware failure the system will fail independently from the software and vice versa. From this fault tree we could argue that the system can fail in a very easy way and safety measures are necessary to avoid a system failure or make it more difficult to occur.

2.8.3 Hazard Analysis for product lines

The product line hazard analysis [41] is slightly different compared with the hazard analysis for single products. For such hazard analysis there are more aspects to be considered, e.g. product line variants, since that for a hazard of a product line member there could be different causes compared with another product line member, e.g. because the system architecture between the product line members is different.
Hence, the product line hazard analysis must also consider all the variants related to the product line members in order to be considered valid for the entire product line. An investigation of several hazard analysis techniques for product lines can be found in [42]. In the next section an FTA approach for product lines is described.

2.8.3.1 Product line Fault Tree Analysis

Product lines are set of similar systems and as such the fault tree analysis used to analyze the causes of hazards for single systems is not enough. Hence, an analysis addressing the variabilities of product lines is needed. A (Software)FTA approach for product lines is presented in [43] and it is briefly described in this section.

To manage product line variabilities in a fault tree, they need to be explicitly related to the event(s) of the fault tree. When a specific variant is present in a product line configuration, then also the related fault tree event(s) should be included in the fault tree. The events related to the product line commonalities instead, will be always present in the fault tree, since they are related to aspects that are always present in all the systems of the product line. Once that the fault tree has been defined for the product line, a refinement process, to make it valid for single systems is needed.
In Figure 20 an example of software fault tree for product lines and the relative refinement for one product is presented.

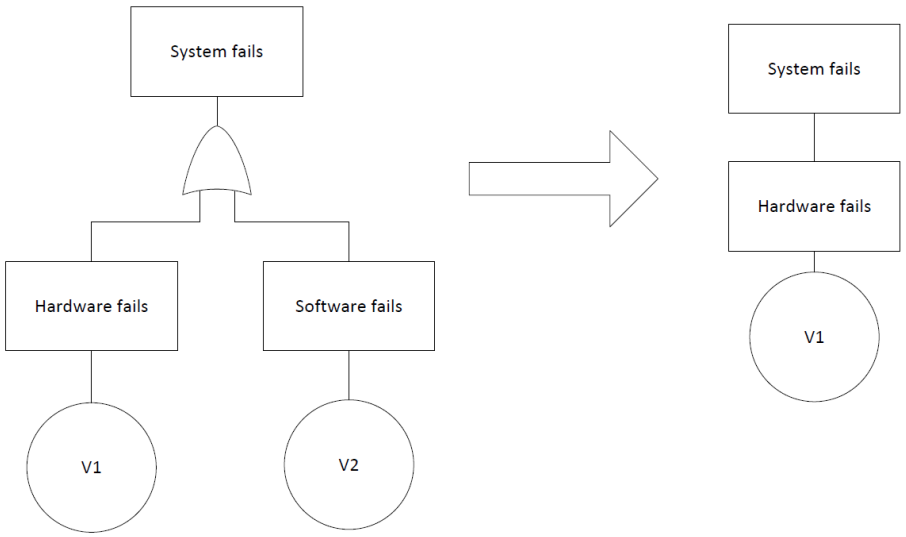

Figure 20: Fault tree for product lines based on the notation used in [43]

For the example of Figure 20 we assume that two variabilities for a product line have been identified during commonality and variability analysis, e.g. V1 and V2. V1 and V2 are related respectively to a hardware failure event and software failure event in the fault tree. When the variability V1 will be present in the product line configuration then, also the hardware failure event in the fault tree will be present. While, if V2 is not present in the product line configuration, the relative event in the fault tree, i.e. Software failure, will be not present. This means that when the variability V1 is in the product line configuration only hardware failures can cause a system failure.

When events related to product line variabilities are removed from the product line fault tree, it could be needed a refinement of the fault tree, e.g. when two events are connected to a logic gate and one of them is related to a variability that is not included in the product line configuration, the relative logic gate has no reason to exists and there can be a direct link between the parent and child event without any logic gate. The result of the refinement for the considered Fault tree is represented in Figure 20.

2.9 Fuel Level Display systems

The *Fuel Level Display System* (FLDS) is a system used in Scania's trucks and buses to estimate and show the fuel level to the driver, and to activate an alarm when the fuel level is low. The FLDS is used both in trucks and buses, and different versions of the system exist.
Scania produces trucks and buses with liquid and gas fuel engine, and to a different vehicle type and fuel type corresponds a different version of the FLDS. This is because, for example, a different tank type or different fuel sensors can be used and these differences have an impact on the FLDS.
In addition to the vehicle type and fuel type there are also other aspects that influence the FLDS. These aspects are: the injection system, fuel volume, number of fuel tanks etc. For simplicity, however, only the vehicle type and fuel type will be considered as aspects that have an influence on the FLDS. The other parts of the vehicle will be ignored, assuming that they don't influence the FLDS. For this thesis only vehicles with one fuel tank will be

considered.
As mentioned above the FLDS has two main functionalities:

1- Showing the fuel level to the driver
2- Alerting the driver when the fuel level is low.

The fuel level is presented to the driver through a gauge and a warning light is used for signaling a low fuel level. Both functionalities are present in trucks and buses with liquid fuel engine and buses with gas fuel engine. While, in trucks with gas fuel engine the low fuel level warning functionality is not present, because of technical constraints.
The FLDS is composed by several *Electronic Control Units* (ECUs) connected each other through a CAN bus, and depending on the priority of the messages sent and received from and to one ECU, it is connected to a different CAN bus. The red can bus is used for messages with high priority and the yellow one for medium-priority messages. For simplification purposes, it will be assumed that all the ECUs are connected through only one CAN bus.

The ECUs composing the FLDS are:

- *Instrument cluster system* (ICL) : it is responsible to set the position of the gauge's needle and activates the low fuel level alarm when needed. For buses it is also responsible to determine if the fuel level is low.
- *Coordinator* (COO): it is the main ECU and it is responsible for the fuel level estimation. For trucks with liquid fuel engine it is also responsible to determine if the fuel level is low.
- *Engine Management System* (EMS): it is responsible to calculate the fuel consumption rate for trucks with liquid fuel engine, while for trucks and buses with gas fuel engine it is responsible to read the fuel level in the tank.
- *Body chassi system* (BCS): for buses with two fuel tanks it is responsible to read the fuel level in the second tank, but since that only buses with one fuel tank have been considered, it only forwards back the received messages.
- BCI1: It is responsible to read the status of the parking brake in trucks with liquid fuel engine.

To a different version of the system corresponds a different ECU configuration. In Figure 21 the system architecture for trucks with liquid fuel engine is presented.

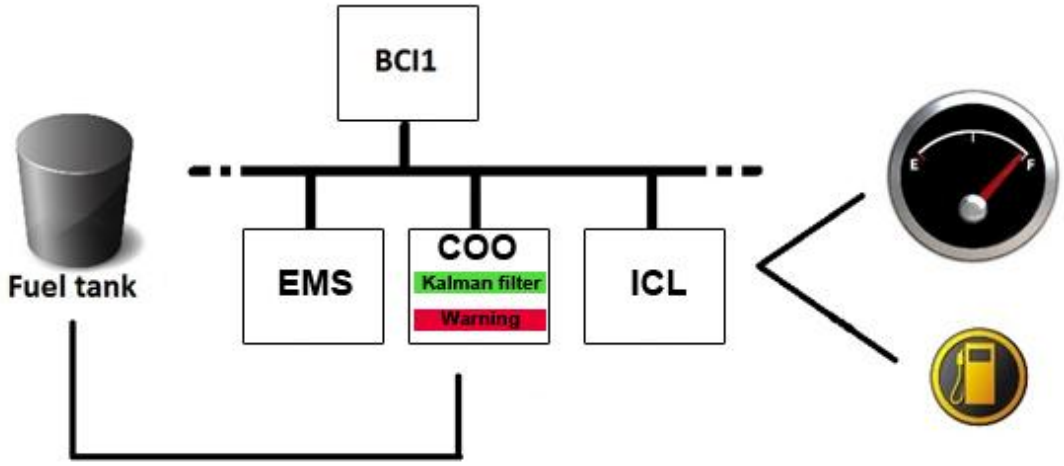

Figure 21: ECUs configuration for trucks with liquid fuel engine

In this type of vehicles the BCI1 reads the status of the parking brake, the EMS reads the fuel consumption rate and they send this information to the COO. The COO reads the fuel level in the tank and estimate the actual fuel level applying a *Kalman filter* to reduce the noise. The information received from BCI1 and EMS is also used in the estimation process. After that the fuel level has been estimated, COO determines if the fuel level is low and send these information to the ICL. Then, the ICL sets the position of the gauge's needle and activates the low fuel level warning if needed, based on the messages received from COO.

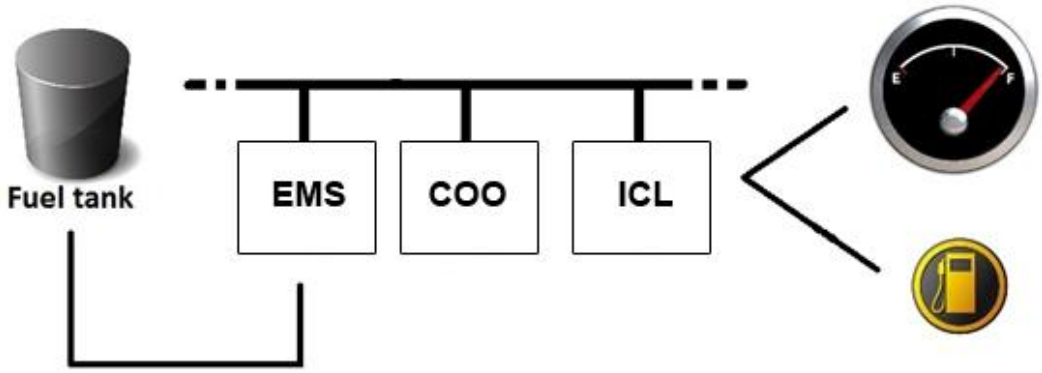

Figure 22: ECUs configuration for trucks with gas fuel engine

For trucks with gas fuel engine there is a different ECUs configuration as we can see from Figure 22. In these vehicles the EMS reads the fuel level in the tank and no filter is applied on the read value. Then, it sends this information to the COO that forwards it to the ICL. At this point the ICL sets the position of the gauge's needle based on the fuel level. In these vehicle the low fuel level warning functionality is not present, because of technical constraints.

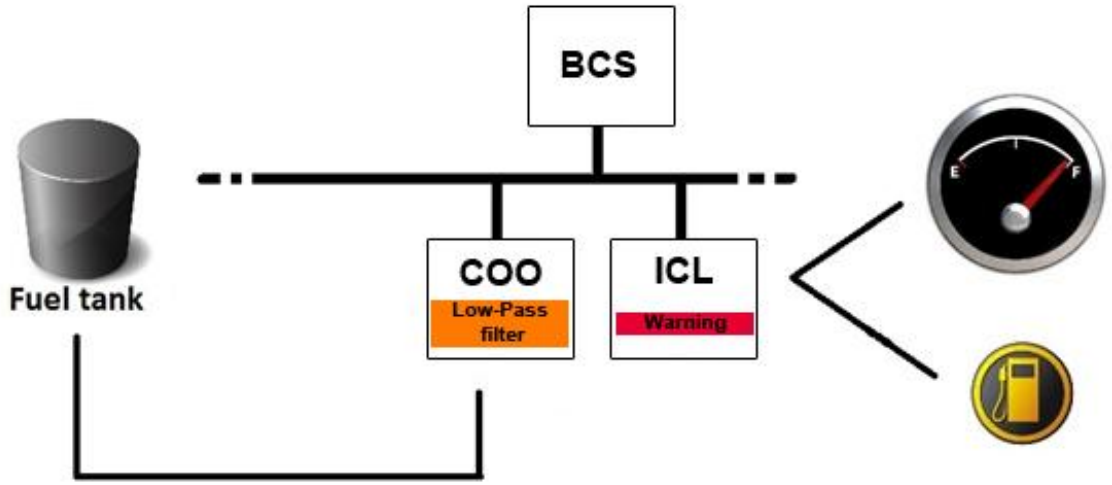

Figure 23: ECUs configuration for buses with liquid fuel engine

The ECUs configuration for buses with liquid fuel engine is depicted in Figure 18. In this vehicles the COO estimates the fuel level applying a *Low-Pass filter* on the fuel level read from the tank, to reduce the noise. Then, COO sends the estimated fuel level to the BCS that forwards it back to the COO. After that, the COO sends the fuel level to the ICL that sets the position of the gauge's needle and determines if the fuel level is low, activating the relative warning if needed.

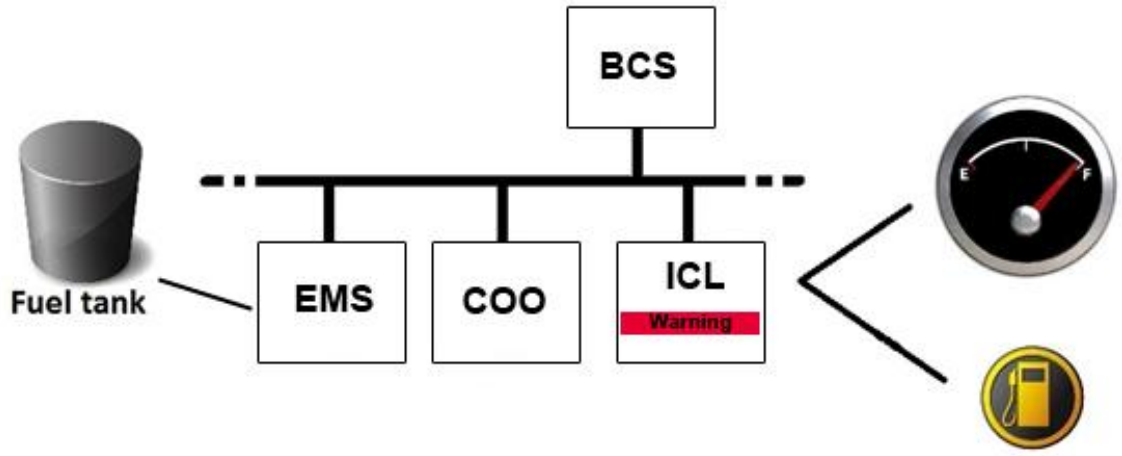

Figure 24: ECUs configuration for buses with gas fuel engine

Finally, the ECUs configuration for buses with gas fuel engine is shown in Figure 24. In this type of vehicles, the EMS reads the fuel level in the tank and sends this information to the COO without applying any filtering. The COO sends the fuel level to the BCS that forwards it back to the COO. Then the COO sends the fuel level to the ICL that sets the position of the gauge's needle and determines if the fuel level is low, activating the relative alarm if needed.

These systems have many common aspects and some differences and thanks to this, the different versions of the system can be managed as a product line, where common aspects define commonalities and differences define variabilities.
This thesis work is based on these systems, considering them as a product line.

2.10 Related work

Some studies have been conducted on adapting PLE method for safety-critical systems and/or enabling reuse in safety certification. In [44], authors focus on several issues involved in PLE, proposing a meta-model to extend PLE for safety critical systems, for which certification is required. The proposed meta-model capture the entities involved in product line certification and the relationships among them, e.g. features, development activities to be performed and the level of certification to be achieved. A model-driven process for the development of Software PLs, ProLiCES [45], has been modified to serve as an example for their approach. This study has been done in the context of the avionic domain.
Authors mainly focus on defining an approach to determine which activities have to be performed and which features have to be included for a certain level of certification. This in order to balance the product cost, including only the necessary activities and features in the product line development process, in the context of a certain level of certification. In this sense they limit their attention to the variability of the processes to perform in developing a safety critical product line.
In this thesis work instead, an approach for developing safety critical product lines, in the context of automotive domain is proposed. The main focus is on defining an approach for evaluating the impact of the variation points configuration within a product line development activity on the other development activities, including the building of the safety case. In more detail the proposed approach shows how a choice within a product line artifact, e.g. Usage context, Feature diagram, Requirement diagram, influence the goal structure that documents a safety case.

As mentioned in Section 2.4.1, a GSN extension has been proposed to build safety cases for safety critical product lines, combined with GSN pattern extensions [26]. In the mentioned approach the authors use the extended GSN to capture safety case variations and trace these variations to the architectural model of the product line. They use as case study a product line of aero-engine control systems.
In this thesis work, we use their GSN extensions to build the safety case for the product line. Furthermore, in addition to capture the relations between the architectural model and the goal structure, the relations between several work products of the activities required by the ISO 26262 and the goal structure are captured. In more detail, an explicit relation between the variation points within each considered work product and the goal structure documenting the safety case is specified. This is done, in order to evaluate the impact of several aspects of a product line configuration on the safety case, not limiting the attention to the architectural model.

In [46] it is discussed how a model based systems engineering approach enables the integration of the safety activities into the development process. It is also discussed how the integration in the product line engineering process with safety activities required for certification, e.g. building of the safety case, could lead to change required for developing

safety critical product lines in an efficient way. In this way, reducing the complexity of developing and managing safety critical product lines. The authors mainly limit their attention to the discussion of the problems related with the integration of safety activities in a model based and product line engineering approaches. Finally they propose a tool supporting the discussed model based approach. The intention of including the definition of the safety case in the product line process is expressed, but no approach is presented.
In this thesis work instead, we provide a approach for the inclusion of the building of the safety case in a product line engineering approach. Furthermore, the considered engineering approach is extended with some of the safety activities required by ISO 26262. Giving in this way the concrete possibility to build the safety case for a safety critical product line, also tracing the relations between the work products variabilities and the safety case ones.

In [47], authors build a safety case for an automotive safety critical system. They build the safety case considering the Functional safety concept clause of the Concept phase section, in the context of ISO 26262. In particular, they examine how model-driven development and assessment can provide a basis for the systematic generation of Functional safety requirements. Furthermore, they demonstrate how a safety case can be traceably developed in relation with the considered models. A case study based on an air suspension system is presented.
The authors build the safety case for one single system, also if some variabilities are identified in the context and feature modes. However, the variations are not considered for the building of the safety case as it is done in this thesis work. In addition to capture the relations between the models and the safety case, in this thesis work the building of the safety case is explicitly done for a safety critical product line and not for one single systems. Showing how a variation point configuration in a work product can influence the building of the safety case. Furthermore the entire Concept phase and part of the Product development at system level sections of ISO 26262 are considering, not limiting the attention to the Functional safety concept clause.

In [48], a similar approach to the one presented in [20] and described above is proposed. They focus on the relation between features, usage context and level of certification to be achieved. In more detail, authors focus their attention on safety critical product lines in the avionics domain. They analyze the relations among the usage context and certification levels to be achieved and how features contribute to the achievement of a certain certification level in a given context.
This thesis work also focuses on PLE for safety-critical systems. However, we focus on the automotive domain and we also consider how the presence or absence of a feature contribute to the building of the safety case. This thesis work is not limited to analyze the features, but also how other aspects of a product line, e.g. hazards, safety requirements, system architecture, influence the safety case.

In [38] an Architecture Description Language for automotive embedded systems, called EAST-ADL, is presented. This language offers constructs for modeling different abstraction levels of the same system, e.g. user features at the top level and system components at the design level. Furthermore, in the context of ATTEST2 [49], these abstraction levels have been aligned with ISO 26262. EAST-ADL extensions are also available for managing variability in the context of product lines. Variability is represented through feature diagrams and variation points. Instead, configuration links are used to keep the traceability among the

different variation points configurations (configuration links are the equivalent of the *obligation* elements used in the approach proposed in this thesis work). However, variability modeling is not supported at each step of the development and safety lifecycle.
This thesis work, instead, aims at guiding (safety) engineers in systematizing reuse during the concept phase and part of the product development at the system level sections of ISO 26262, in the context of product lines development. Specific product lines techniques are indicated for each step, in order to manage variability for each activity required by ISO 26262, including safety case building.

3. Problem formulation and analysis

In this chapter the problem to be solved is described in detail and it is analyzed in order to identify sub-problems to ease its resolution. This chapter is organized as follows: In Section 3.1 the problem to be solved is presented in detail; In Section 3.2 the problem is analyzed and it is decomposed into less complex sub-problems.

3.1 Problem formulation

For safety-critical product lines operating in domains regulated from safety standards, in addition to build them in an acceptably safe way, it must be shown that they are actually acceptably safe. In the automotive domain and, more specifically, in the context of ISO 26262, to show that a system is acceptably safe a safety case must be provided. This safety case is expected to show that the process defined in the safety standard has been followed and that the system behaves in a safe way.

Since a product line is composed by several products, showing that it is acceptably safe, requires the provision of a different safety case for each product composing the product line. Since building a safety case is a time consuming, then costly activity, this approach could be too expensive for companies. Losing in this way many advantages of a product line approach.

Product line engineering process focuses on defining all the aspects of the product line in such a way that a single product can be built in a systematic way, supporting reusability and saving time and costs. Since reuse as well as time and cost reductions are crucial also in the context of safety management, a similar process could be used concerning safety-related aspects, including the building of safety cases. Since products of a product line exhibit many commonalities with some variabilities, the way of arguing their safety could also be common with some variations. Instead of building the safety case from scratch for each single product of a product line, a method that allows safety managers to build the safety case in a systematic way, should be adopted.

Defining commonalities and variabilities, in the way of arguing safety, requires that the commonality and variability analysis has also to be performed for other aspects of the product line, such as safety requirements, hazard analysis, system architecture etc. But in Scania, the different versions of the Fuel Level Display System are not managed as a

product line, and the commonality and variability analysis has not been performed. Due to this, the common and variable aspects have not been explicitly identified.
Hence before building the safety case, in order to define what varies and what remains constant among the different versions of the Fuel Level Display System, the commonality and variability analysis has to be performed. In addition, this analysis has to be carried out taking into account the activities required from ISO 26262, and the common and variable aspects must be organized and modeled using a product line approach. Furthermore, some of the available product line approaches could also not be suitable for this thesis work, hence some adjustments could be required. Also the traceability among variation points has to be kept. This is, in order to keep coherently configure the different variation points within the whole product line. Furthermore, also some tools supporting the selected product line approaches have to be used.

Finally, the main goal is to define a configurable safety case that is valid for the whole product line, hence that is based on its commonalities and variabilities. In this way, allowing the systematic building of the safety case for each product composing the product line in order to ease the product line development and certification.

3.2 Problem analysis

In order to solve the problem presented in Section 3.1, it has been decomposed into less complex sub-problems, each of them covering a different aspect of the original problem.

3.2.1 Problem scope

Due to the limited amount of time available for this thesis work, it is important to define the scope of the problem. To do this, first of all a study of the ISO 26262 safety standard and the FLDSs has to be done, in order to take confidence with them.
Since the FLDSs in Scania for many aspects are not managed as a product line, an investigation has to be done in order to understand which information is available for all the versions of the system. Furthermore, this investigation has to be done taking into account which information is required from ISO 26262. Since, due to the limited amount of time, building a safety case covering all the aspects of the safety standard is not feasible, the parts to cover in the standard have to be defined. Hence, while investigating which information is available for the different versions of the FLDSs, a mapping between the information available for the different versions of the FLDSs and the information required from the parts that will be considered of ISO 26262 has to be done.
Once that the scope of the safety standard and the information to gather for the FLDSs have been decided, it is possible to proceed with a commonality and variability analysis.

3.2.2 Commonality and variability modeling approaches

For representing commonalities, variabilities and constraints on variability selection, for the FLDSs, several product line techniques might to be used. Hence, a review of the literature is needed to find suitable approaches to model the common and variable aspects of the

systems that have been considered, e.g. requirements, system design etc.
After this investigation, if more approaches are available to model the same aspect, only one has to be chosen, based on its advantages and disadvantages in relation to this problem. Once that one approach for a considered activity has been selected, if it presents limitations in relation to this problem, a modification or an extension has to be proposed, in order to cover the lack. In addition, a valid modeling tool has to be chosen in order to manage the representation of the different models.

3.2.3 Commonality and variability analysis

As mentioned in Section 3.1, the different versions of the FLDS are not managed as a product line. Hence, a commonality and variability analysis among all the version of the system has to be done. The commonality and variability analysis has to be performed considering the information that has been previously gathered in accordance with ISO 26262. Hence, for each activity that will be considered a commonality and variability analysis has to be performed in order to treat the different versions of the FLDS as a product line. Furthermore, in addition to identify commonalities and variabilities for the FLDSs, also the relations among them have to be defined. Since there could be explicit connections between variation points, e.g. one variant could influence the configuration of a different variation point, constraints have to be explicitly represented.
Furthermore, it could be necessary to limit the amount of information to consider for the FLDS during the commonality and variability analysis. Hence, also this aspect has to be considered, in order to perform the thesis work in the range of time available.
Once the commonalities, variabilities and the possible constraints on the variability selection have been identified, it is necessary to model them, using the chosen product line techniques.
After that commonalities, variabilities and constraints on the variability selection for the FLDSs have been identified and modeled, the different versions of the system can be considered as a product line (Fuel Level Display Product Line (FLDPL)).

3.2.4 Safety case building

Once the FLDPL has been explicitly defined, it is possible to proceed with the building of the safety case. The safety case has to be defined taking into account the commonalities and variabilities of the product line. Meaning that commonalities and variabilities have to be defined also within the safety case. Furthermore, the identified variants have to be selected according to the configuration of other variation points within the product line. For example, to a certain configuration of a variation point in the system design can correspond a specific way or arguing safety within the safety case.
Hence, an explicit mapping between the variation points for the different aspects of the product line (e.g. system design) and the variation points within the safety case have to be defined, in order to support the safety case configuration based on single systems of the FLDPL.

4. Method

In this chapter the product line approaches, useful to model and or analyze several aspects of a product line, are presented. More specifically, in Section 4.1 the usage context is considered; in Section 4.2 requirements are took into account; in Section 4.3 the product line behavior is handled; In section 4.4 hazard analysis is considered; in Section 4.5 the product line design is treated and finally in Section 4.6 the safety case is considered.

In order to be ISO 26262-compliant the product line engineering process has to be aligned with the safety standard, since ISO 26262 does not contains guidelines for product lines. This means that each activity of the ISO 26262 has to be first performed for the entire product line and then through a configuration process the relative version for single systems can be derived. So far in the considered literature, a complete alignment of a product line engineering process with ISO 26262 has not been done. Hence, product line techniques for the activities that have to be performed, have to be found among the ones available so far.
First of all, the parts of the standard to be covered have to be defined and after that, each single required activity has to be taken into consideration. For this purpose, this thesis focus on the third section and part of the fourth section of ISO 26262. Section 3 corresponds to the Concept phase (Section 2.2.1), and Section 4 corresponds to the Product development at system level, (Section 2.2.2).
The choice of these two phases is also motivated by the fact that with respect to them the information available for the FLDSs highly matches the one requested.
However, the activities required from the safety standard, within the considered sections, for which there were no information available, have not been considered in the scope of the thesis work.
More specifically, the clauses that are considered within the Concept phase are the Item definition, Hazard analysis and risk assessment and Functional safety concept clauses. Instead, within the Product development at the system level section, the clauses that are considered are the Specification of the technical safety requirements and System design clauses.
The aspects of the systems that has been taken into account are the usage context, functional requirements, system behaviour, hazard analysis, safety requirements and system design.
Since the FLDSs were not managed as a product line in relation with the mentioned aspects, a product line approach for each of them has been chosen among the ones available in the considered literature. In the next sections for each aspect that has been considered the relative selected product line approach is presented.
Since some of the available product line techniques didn't fully satisfy the case of the FLDSs, limitations has been highlighted and the proposed extension or modification has been presented in Chapter 5.
From now on, the FLDSs will be intended as a product line, and as such they will be referred as Fuel Level Display Product Line (FLDPL).

4.1 Product line usage context

As explained in Section 2.6, Feature diagrams are suitable for representing product lines from a feature viewpoint. These diagrams can also be used to model the usage context of a

product line as shown in Section 2.6.1.
This method is totally suitable to model the usage context of the FLDPL without the need of any extension or modification. In addition other methods to represent the usage context of a product line, in the considered literature, have not been found. Hence, the method presented in Section 2.6.1 is the only candidate to model this aspect of the FLDPL. For this reason it has been selected for this thesis work.
An investigation has also been done in order to find a usable tool to create feature diagrams.
Three plugins for Eclipse [50], supporting feature diagrams, have been found: EMF Feature Model [51], Feature diagram editor [52] and Feature IDE [53]. But, for practical reasons and familiarity with Microsoft Visio, new stencils have been provided for modeling feature diagrams. The proposed solution is presented in Section 5.2.

4.2 Product line requirements

When representing requirements for product lines, variation points and variants have to be considered [54]. Hence methods for single systems are not suitable and they need to be extended.
An approach for developing and representing requirements for a product line is shown in [55]. This approach proposes the usage of a tool (DREAM). However, for practical reasons and since the main focus of this thesis is not to investigate tools supporting product line requirements, it has not been selected for this thesis work.
A simple way of representing requirements for a product line is also present in [43]. It is based on a textual approach, grouping common and variable requirements in different sections. In addition another section for dependencies (inclusion, exclusion) among requirements is considered. Although this is a very clear way of representing requirement for a product line, it has been preferred to find a graphical representation.
This is because currently in Scania, a tool for representing requirements for single systems is being developed, and the requirements are represented through a requirements tree. Hence, in order to follow the method used by Scania, it has been decided to represent requirement for the product line using a tree-like structure.
Instead of introducing an additional notation or extending an available approach for representing requirements for a product line, it has been found that feature diagrams (Section 2.6) could be used for this purpose. Instead of using this notation for describing a product line from a feature viewpoint, it can be used to represent the requirements of a product line without the need of any extension.
In this way it is possible represent requirements in a tree, following the method actually used by Scania. In addition, feature diagrams give the possibility to trace requirements among them, showing how a requirement is decomposed in sub-requirements. This could result in a really usable mean for managing traceability among requirements.
For these reasons feature diagrams have been chosen for representing requirements for the FLDPL, in this thesis work.
As shown in Section 2.6.1 there can be intrinsic and extrinsic constraints between features, represented through a constraints table. In the same way there can be intrinsic and extrinsic constraints between requirements. In order to simplify the comprehension of the feature diagram, it could be helpful to represent these constraints within the feature diagram itself in a graphical way, instead of having a separate table for the constraints. For this reason an extension for feature diagrams has been proposed in Section 5.3.

Since feature diagrams are also used to represent the usage context, the same tool can be used for this case. The only difference is that the tool should support the mentioned extension. As already explained in the previous section the tool that will be used for representing feature diagrams is shown in Section 5.2. How the proposed extension has been managed, in the adopted tool, is also described in Section 5.3.

4.3 Product line behavior

Activity diagrams represent a common means to model the system behavior (Section 2.7.1). These diagrams are usually used to represent the behavior of single systems. Constructs for representing variability within activity diagrams are the decision nodes with parameterization. However, they are not used to represent variability related to product lines.

A method to express variability for product lines, within activity diagrams has been proposed in [56]. It uses decision nodes to implicitly define variation points, and for each relative variant a mapping with the relative feature is expressed. Anyway, it is not really clear when a decision node is relative to a variability for a single system or a product line.

Another method has been proposed in [36]. However, it has not been deeply investigated since the one presented in [56] has been retained really close to what was needed for this thesis work.

Since a complete and intuitive mean for representing variability within activity diagrams, for product lines, has not been found, it has been decided to propose a new approach based on the considered ones. This new approach is presented in Section 5.4.

Concerning the tool for managing activity diagrams, Microsoft Visio has been used. This tool offers all the constructs for managing activity diagrams, and it is commonly used in Scania to manage different type of diagrams. For this reasons it has been chosen for this thesis work. In addition it has been easily used also for modeling the proposed approach. This is also shown in Section 5.4.

4.4 Product line hazard analysis

As described in Section 2.8, Hazard analysis techniques can be inductive, deductive or hybrid. In this section two types of hazard analysis techniques are treated. In more detail in Section 4.4.1 hybrid hazard analysis is considered and in Section 4.4.2 deductive hazard analysis is treated.

4.4.1 Hybrid hazard analysis

An hybrid hazard analysis technique is the Hazard and operability analysis (HAZOP), as described in Section 2.8.1. This technique has been also extended to comply with ISO 26262 adding four more columns (Controllability, Exposure, Severity and ASIL). In addition, in [2] it has already been performed for one version of the Fuel Level Display System. For this reasons this technique has been selected for this thesis work.

However, since it is suitable only for single systems and since in the consulted literature, no extension for product lines has been found, an extension to make this technique suitable for product lines has been proposed. This extension is described in Section 5.5.

4.4.2 Deductive hazard analysis

A deductive hazard analysis technique is the Fault Tree Analysis (FTA), as described in Section 2.8.3.1. This technique has also been extended for product lines and its usage is also suggested by ISO 26262. These reasons have been retained enough to choose this technique for this thesis work.
However, in the product line version of the FTA, the fault tree symbols are not used as for the version for single systems, i.e. the symbol for basic events are used to represent conditions and basic events are represented using intermediate events symbol. This can lead to confusion, and no justification is provided for this change of notation. For this reason a light modification to this approach has been proposed and described in Section 5.6.

Concerning the tools supporting the fault tree for product lines, something has been found in the consulted literature. PLFaultCAT is an interactive, partially-automated support tool to aid software engineers in the application of product-line software SFTA [57]. However, for practical reasons and since the author was more familiar with other tools, PLFaultCAT has not been selected for this thesis work.
Microsoft Visio, in addition to give the possibility for creating activity diagrams, also support Fault Tree modeling. Since this tool has already been chosen to model other aspects of the product line, e.g. activity diagrams, it has been preferred to PLFaultCAT. In this way it has been possible to don't introduce a new tool for modeling this aspect of the product line.
Unlikely, Microsoft Visio doesn't support Fault Tree modeling for product lines, but a simple solution for this limitation has been proposed in Section 5.6.

4.5 Product line design

Several approaches to model the design of product lines have been proposed. In [58] an approach has been proposed, but it is mostly related to software design. In [29] approaches to model different aspects of a product line (including system design) have been described. However, it was missing the constructs to represent interfaces and information flow between components. Hence, it has been abandoned. In [59] another approach has been proposed, but it is not specific for system design.
The approach that has been retained more complete for representing product line design is the one presented in [60] and in Section 2.7.3. With this approach, using SysML block definition and internal block diagrams, it is possible to model the product line design, all the interfaces between the elements composing the product line and also the information flow. In addition it also is possible to keep a tracking between the feature selection and the impacts on the product line design. For these reasons it has been retained a valid approach, and it has been chosen for this thesis work.

Furthermore, a plugin for Microsoft Visio [61] is available to create these models. There hasn't been the need of an investigation for other tools supporting SysML, since Microsoft Visio has been retained a valid one. In addition, Microsoft Visio has also been chosen for creating activity diagrams, hence avoiding the usage of a different tool for modeling a different aspect of the product line.

4.6 Safety case line

Building a safety case for a product line is different rather than building a safety case for a single system. In the case of a product line, a monolithic safety case is not enough to show the product line acceptable safety, since the product line is composed by different members and each member could require different product and process based evidences.
Hence, to solve this problem, a modular and configurable safety case should be built, in a way that it can be adapted to each member of the product line. Meaning that, to one configuration of the product line corresponds the relative configuration of the safety case.
Defining a good safety case architecture, is the base to reach this goal. A good mean to represent safety case architecture is GSN, as described in Section 2.4.1. GSN provides modeling capabilities for single systems and also extensions for modularity and product lines. For these reasons it has been chosen for this thesis work, and it will be used to define a goal structure that is valid for all the members of the product line.

For creating a goal structure with GSN it is possible to use several tools. Some of the available ones implementing the GSN are listed at [62], but not all of them are free to use. One free resource is the plugin developed for Microsoft Visio, but it has been found that GSN product line extensions were not supported. Hence, it has not been selected for this thesis work.
Also a plugin for Eclipse supporting GSN has been developed and its name is Assurance Case Editor [63]. This plugin has been easily installed but, as for the Microsoft Visio plugin, it does not support the extensions for product lines and its usage has been abandoned.
Hence, a free tool supporting GSN extensions for product lines has not been found. For this reason it has been necessary to manage this aspect in a different way. A solution has been proposed in Section 5.7.

5. Methodological and modeling support for building safety case lines

In this chapter the product line approaches limitations highlighted in chapter 4 are addressed. In more detail in Section 5.1an overview of the proposed approach is presented; in Section 5.2 a means to easily draw feature diagrams is proposed; in Section 5.3 a feature diagram extension is explained; in Section 5.4 an approach for extending activity diagrams is described; in Section 5.5 and adapted version of HAZOP analysis is presented; in Section 5.6 an adjustment for product line FTA is described; finally in Section 5.7 a tool supporting GSN is treated.

5.1 Approach overview

As mentioned in Chapter 4, ISO 26262 does not contains guidelines for product lines, and an alignment of the product line engineering process with the safety lifecycle specified by ISO 26262 is necessary. For this purpose, all the product line techniques presented in Chapter 4

and the relative proposed adjustments in this chapter, are aimed to be coherently used together towards the provision of a method for building a safety case for ISO 26262-compliant product lines. The proposed product line engineering process aligned with ISO 26262 is summarized in Figure 25 and extends the one presented in Section 2.5, that is constituted of two phases: domain engineering and application engineering phases. This process is also extensively explained in [4], where it has been named VROOM (Vehicle Road Obligation Oriented Method) & controlled CRASHES (Claim Reuse for Arguing Safely on Hazards Elimination and or Softening).

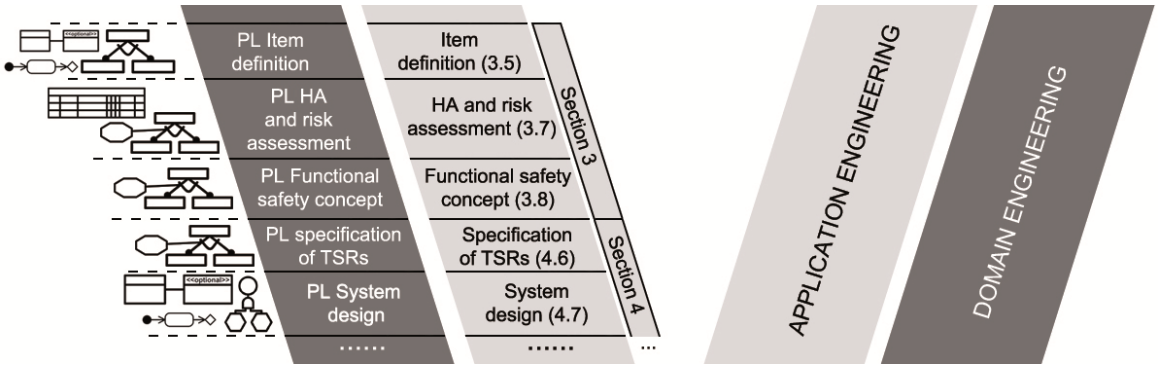

Figure 25: ISO 26262 safety life cycle and PLE alignment

In more detail, for each section and clause of ISO 26262, the first phase to be performed is the domain engineering one, where the focus is on the entire product line. And after that, the application engineering phase is performed, where the focus is on the single product. In this way, each activity required from ISO 26262 is first performed on the entire product line, and through a configuration process, the relative version for one single product is derived. To do that, appropriate product line techniques have to be selected for each level of the process.

In more detail during the Item definition, feature diagrams to represent the usage context and the requirements are adopted; SysML block definition and internal block diagrams are used to represent the preliminary architecture; and activity diagrams are adopted to describe the behavior of the item. Successively, during the Hazard analysis and risk assessment an adapted version of the HAZOP analysis is used, and feature diagrams are adopted to represent safety goals. Feature diagrams are also used to represent functional and technical safety requirements, and to keep the traceability among safety requirements. Finally during the System design, SysML block definition and internal block diagrams are used to represent the product line architecture, activity diagrams to represent its behavior, and fault tree analysis is used as safety analysis technique on the system design.

All the activities of the proposed product line engineering process are not independent each other. Thanks to the possibility to represent variation points at each level of the process, and to define constraints on the variants selection for each variation point, it is possible to keep the traceability and coherence among all the variation points of the product line. For instance, a specific configuration of a variation point within the Item definition, could influence the configuration of another variation point within the system design. This influence between variation points is specified through the definition of constraints on the variants selection for each variation point. In this way when a variation point is encountered, it can be configured

taking into account also other variation points, in this way coherently configuring the entire product line.
The process represented by the double V in Figure 25 is flanked by another double V (Figure 26), that is related this time to the safety case lifecycle.

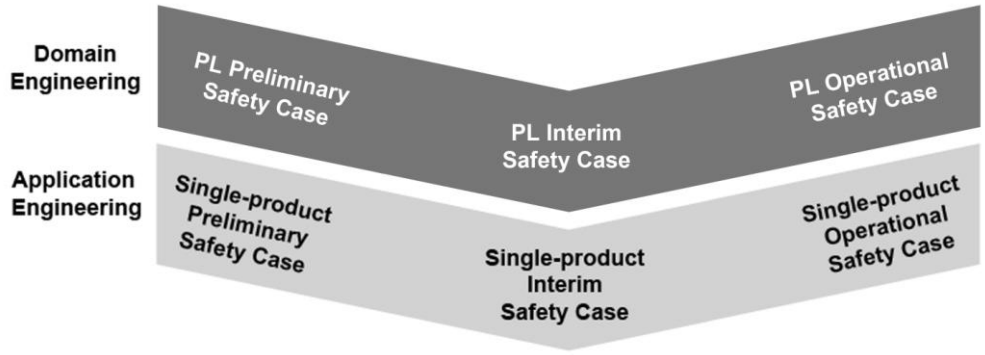

Figure 26: Product line safety case lifecycle

As for all the activities mentioned so far, the safety case building has to be first performed taking into account the entire product line and then through a configuration process, deriving the version for one single product. The different stages of the safety case (preliminary, interim and operational) are represented through GSN and its extensions for product lines. Since GSN gives also the possibility to represent variation points and the relative constraints through the obligations, the goal structure can be configured in such a way that reflect the configuration of the other variation points within the product line. Hence, a different version of the goal structure and the safety case can be derived from a different configuration of the product line.

5.2 Feature diagram tool

As described in Section 4.1, for practical reasons and familiarity with Microsoft Visio it has been decided to use this tool for creating feature diagrams.
Microsoft Visio supports the creation of personal libraries of symbols, named stencils. And this characteristic has been used for the creation of a set of symbols for representing all the elements in a feature diagram. In more detail, five symbols have been created. One symbol representing a feature; one representing a mandatory relation; another one representing an optional relation; a symbol representing a link between features (a solid line), that can be used together with a U-shaped line symbol for representing an alternative relation. The Visio stencil is represented in Figure 27.

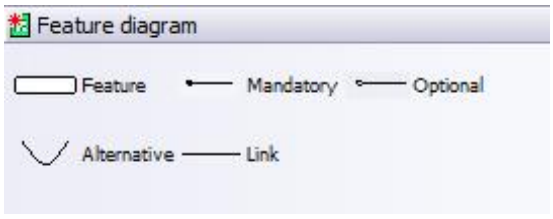

Figure 27: Visio stencil for creating feature diagrams

In this way it has been possible to use a tool, that has been already chosen for modeling also other aspects of the system, e.g. activity diagrams and system design, for managing the feature diagrams. Keeping also in this way, the usage of a unique tool for modeling all the aspects considered so far for the product line.

5.3 Feature diagram obligations

As explained in Section 4.2, in order to easy the comprehension of the feature diagrams, it could be helpful to represent the constraints on the variability selection, within the feature diagram itself in a graphical way. This is in order to avoid to have a separate table for these constraints.
For this purpose, an explicit inspiration from GSN (Section 2.4.1) has been taken. In GSN for modeling these constrains an hexagon symbol is used, named Obligation. It has been decided that it was a good mean for the graphical representation of constraints, and that it could be adopted also in the context of the feature diagrams.
An example of the usage of the Obligation for feature diagrams is shown in Figure 28.

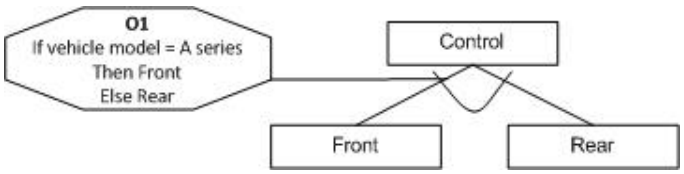

Figure 28: Obligation in the context of a feature diagram

As Figure 28 shows, the Obligation is used to describe a constraint related to a variation point in a feature diagram. In this case if the vehicle model is *A series* then the feature selected will be *Front* else *Rear.*
In this way it is possible to derive a valid feature diagram configuration, without the need of consulting a different tables containing constraint on the feature selection.

This extension for the feature diagrams has been easily integrated in the tool used to create them. As explained in the previous section, a personal stencil in Microsoft Visio has been created for managing feature diagrams. For supporting the introduced extension, a new symbol (octagon shape) has been added to the stencil.

5.4 Activity diagrams for product lines

As described in Section 4.3, since an easy and intuitive means for representing variability within activity diagrams, for product lines, has not been found, it has been decided to propose a new approach based on the considered ones.
Since a way of representing variability already exists, as shown in Section 2.7.1, it has been decided to use the same approach, but adding a new stereotype for distinguishing between a variability related to a single system and a variability related to a product line.
Simply, adding the stereotype <<*variation point*>> close to the decision node, representing

the variability it is possible to understand if it is related to a variability of a product line or not.
The decision of using decision nodes as variation points for the product line has been taken from [56], and the decision of adding the stereotype <<*variation point*>> from [36].
An example of the proposed approach, is shown in Figure 29.

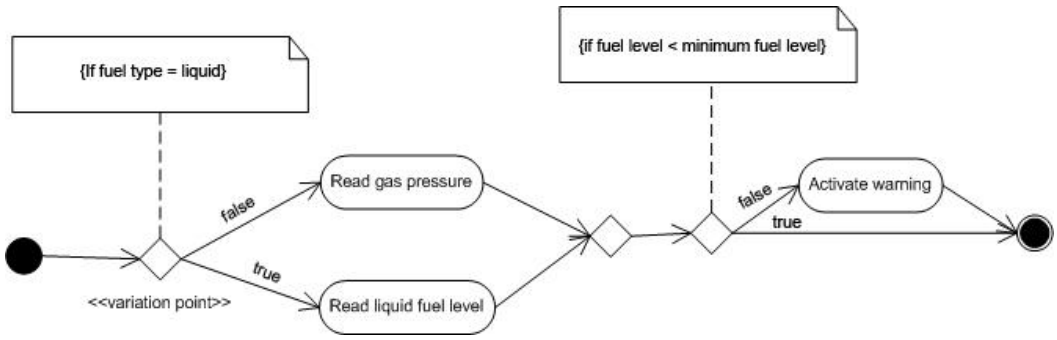

Figure 29: Example of variability for product lines in an activity diagram

As Figure 29 shows, the first activity that is done in the activity diagram, is reading the gas pressure of the fuel level, depending on the fuel type. The first decision node, since it is marked with the stereotype <<*variation point*>> it is relative to a variability of the product line, i.e. fuel type. Then after that the gas pressure of fuel level has been read if the value read is low a warning is activated. This time the decision node is relative to a variability not related to the product line, since there is not the relative stereotype.
Hence with this extension, it is possible to distinguish if a decision is related to a product line variability or not.
Since Microsoft Visio has been used to create activity diagrams, it has been really easy to add a new stereotype for activity diagrams. It has been simply done adding simple text on the bottom of a decision node.

5.5 Adapted HAZOP analysis

As described in Section 4.4.1, HAZOP analysis has been chosen for this thesis work. It has also been extended for complying with ISO 26262 but it doesn't support product lines. For this reason it has been decided to add two additional fields in the HAZOP table presented in Table 7. The fields that have been added are named *Obligation* and *Variation Points*. In the *Obligation* field, several conditions can be specified for the relative hazard. These conditions specify which variant to chose for each variation point within the entry of the adapted HAZOP table. The conditions can also define if an entry of the table is valid for a given product line configuration.
Instead, the *Variation points* field, contains the description of each variation point, represented through brackets, within the relative hazard. Each variation point can have several properties that can be specified in the following style:
VariationPointId: Property1 = {Value1, Value2…}, Property2 = {Value1, Value2…}…

In Table 8 the proposed adapted HAZOP table is presented.

ID	Obligation	Item /interaction	Attribute	Guide word	Deviation	Consequences	**E**	**C**	**S**	**ASIL**	Causes	Variation Points

Table 8: Example of adapted HAZOP table

This adapted HAZOP table, in addition to comply with ISO 26262, also supports product lines, making it compatible for this thesis work.

5.6 Adapted FTA

As explained in Section 4.4.2 in the FTA for product lines presented in Section 2.8.3.1, there isn't a correct usage of the fault tree symbols. For this reason it has been decided to keep the notation used in Section 2.8.2, and to introduce a new symbol for representing a condition. Hence, it has been decided to introduce an hexagon shape to represent conditions in the fault tree. The hexagon shape has never been used for fault trees and it is coherent with the obligation symbol already introduced for the feature diagrams. An example of this adjustment in the fault tree notation is given in Figure 30

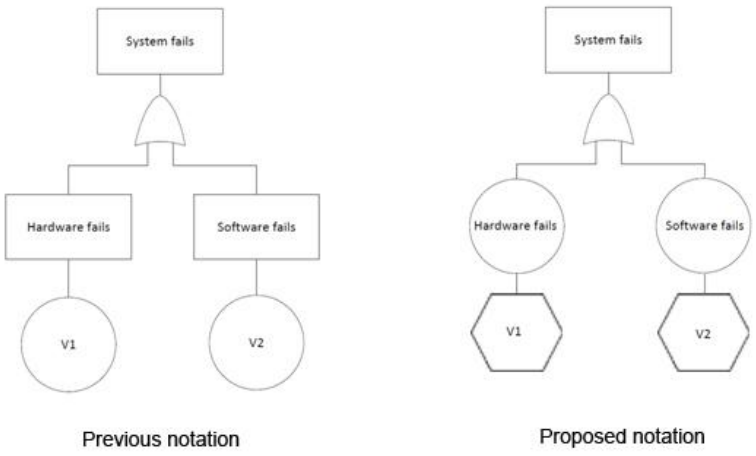

Figure 30: Adjusted notation for product line FTA

As Figure 30 shows instead of representing basic events with rectangles, round shapes are used. And, instead of representing conditions with round shapes, hexagon shapes are used. In this way the same notation is kept for FTA for single systems and FTA for product lines.

As described in Section 4.4.2, Microsoft Visio has been chosen for creating the fault tree. The only limitation that has been found, is that the hexagon symbol was not available in the stencil provided by Microsoft Visio for the fault trees. For this purpose, since the hexagon symbol has already been included in the stencil for managing feature diagrams, the symbol in the mentioned stencil has been used in combination with the stencil provided by Microsoft Visio for creating feature diagrams. In this way it has been possible to model the fault tree for the product line without any additional effort.

5.7 GSN tool

As described in Section 4.6, the available tools supporting GSN don't offer the possibility to use the GSN extensions for product lines. For this reason it has been decided to adopt a similar solution to the one used for the feature diagrams. That is, creating a new stencil in Microsoft Visio, containing all the symbols necessary for creating a goal structure with GSN, including all the ones related to product lines, that were missing in the investigated tools. The new stencil is represented in Figure 31.

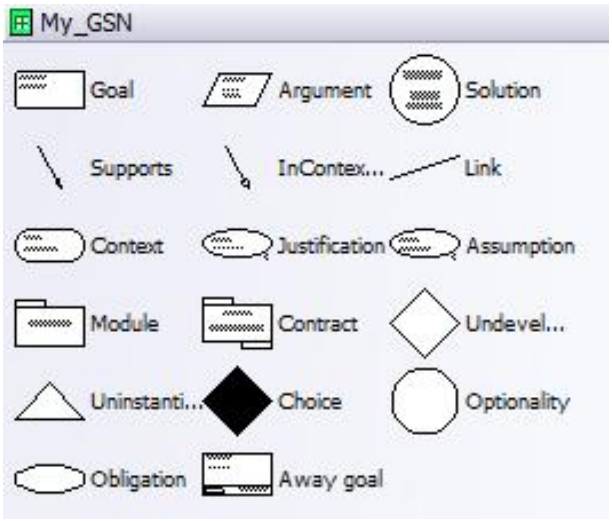

Figure 31: Visio stencil for creating goal structures

As Figure 31 shows, seventeen symbols have been created within the new Visio stencil. These symbols are exactly the ones that have been introduced in Section 2.4.1, and can be used to create goals structures also for product lines.

6. Case study

In this chapter the product line approaches described in Chapter 4 and Chapter 5 are applied on the FLDPL. These approaches are used to describe the product line in accordance with the requirements of ISO 26262. In the first part of this chapter the Concept phase section is considered. More specifically in Section 6.1 the Item definition clause is considered; in Section 6.2 the Hazard analysis and risk assessment clause is took into account; in Section 6.3 the Functional safety concept clause is treated. In the remaining part of this chapter the Product development at the system level section is treated. In more detail in Section 6.4 the Technical safety concept clause is examined; in Section 6.5 the System design clause is considered. Finally in Section 6.6 the safety case for the FLDPL is defined.

6.1 Item definition

As recalled in Section 2.2.1, the goal of the item definition clause is to define and describe the item, its dependences on, and interaction with, the environment and other items. The item in this case is the FLDPL and it is defined in this section. More specifically in Section

6.1.1 the usage context of the product line is presented; in Section 6.1.2 the main functionalities are described; in Section 6.1.3 the product line behavior is explained; in Section 6.1.4 the preliminary architecture is treated; finally in Section 6.1.5 the functional requirements are took into account.

6.1.1 Usage context

Scania produces trucks and buses, and in each of these vehicles a fuel level display system is installed. Hence, the vehicle type can be considered as the environment in which the FLDPL is used. In addition to this, all the parts that compose the vehicle can be considered as its external environment or its usage context (Section 2.6.1).
Hence, the usage context of the FLDPL is defined from the vehicle characteristics and vehicle parts that can affect directly the variability selection of the product line, e.g. vehicle type, number of fuel tanks, fuel type etc.

In order to define the usage context, an analysis of the aspects influencing the variability selection within the product line has been done. The identified aspects are:

1. Vehicle type
2. Fuel type
3. Injection system
4. Fuel tanks number
5. Fuel volume

For each of these aspects different options are available. Since in the real case, the possible options for some of the aspects were quite numerous, it has been decided to consider only a sub-set of them. In accordance with Section 4.1, these information, describing the usage context, has been represented using a feature diagram, which is shown in Figure 32.

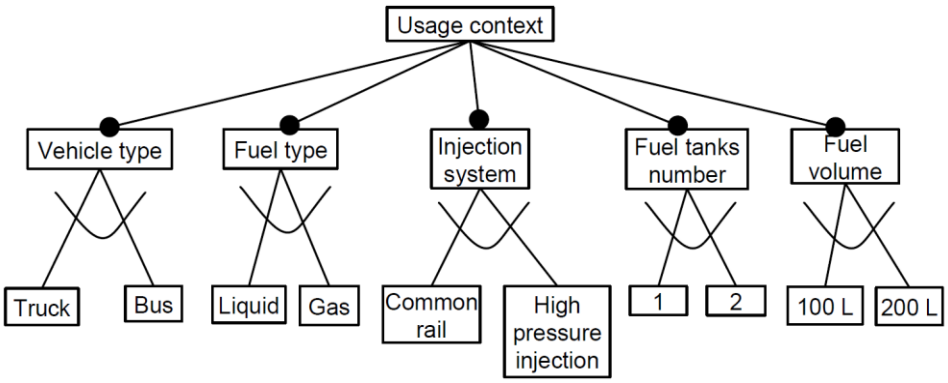

Figure 32: FLDPL usage context

In the presented feature diagram there are five aspects affecting the variability selection of the product line. For each of them there are two possible choices. Concerning the vehicle type, as already mentioned in the beginning of this chapter, trucks and buses are produced by Scania. The selection of one of these choices exclude the other one, since as we can see from the diagram, they are related through a xor-alternative relation. This is logical, since there cannot be a vehicle that can be a truck and a bus at the same time.

Concerning the fuel type, also two possible alternative choices are available. These choices indicate that the engine of a vehicle can be alimented by a liquid fuel or a gas fuel.
The same thing is for the Injection system. Two alternative options are available, that are Common rail and High pressure injection.
In addition trucks and buses can have one or two fuel tanks, and each of the fuel tank can contain one or two hundred liters of fuel.
Hence, in total, there are 16 possible configurations of this diagram. As already mentioned in Section 2.9, due to the limited amount of time, the aspects that have been considered for this thesis work are only the vehicle type and fuel type. Concerning the injection system, fuel tanks number and fuel volume, it will be assumed that all the vehicles have a Common rail injection system, and one fuel tank with a capacity of one hundred liters.

Thus, as represented in Figure 33, the usage context considered, will be composed by the vehicle type and fuel type features.

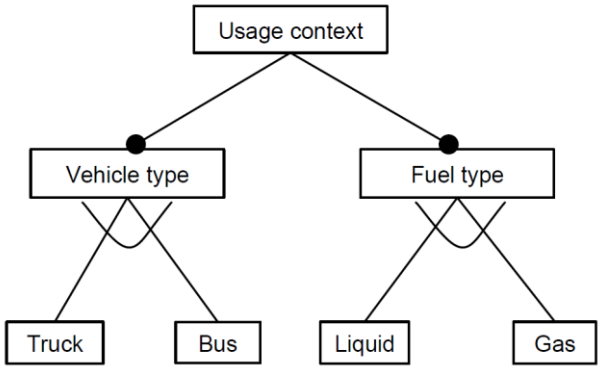

Figure 33: Usage context considered for FLDPL

For this diagram, there are four possible configurations:

1. Truck with liquid fuel type
2. Truck with gas fuel type
3. Bus with liquid fuel type
4. Bus with gas fuel type

For each of these configurations, there is also a different configuration of the FLDPL. In the rest of Chapter 6, it will be shown how each configuration of the diagram in Figure 33 will be the main driver for the variability selection for the other aspects of the product line, e.g. functional requirements, hazard analysis etc.

6.1.2 Functionality

As described in the Section 2.9, the FLDPL has two main functionalities: Fuel level estimation and display and Low fuel level warning.
The first one is related to the estimation of the fuel level in the tank and the visualization of this value, through a gauge, to the driver. While, the second one is related to the visualization of an alarm to the driver, to advise him in case of a low fuel level in the tank.
Reading Scania documents and interviewing the responsible of the product line, it has been found that the Low fuel level warning functionality is present on all the vehicles, with the

exception of trucks with gas fuel type. This type of vehicles doesn't have this functionality because of technical constraints. These constraints will be not presented, because they are confidential information.
In this thesis work we are considering the Fuel Level Display as a product line, but from a vehicle viewpoint it can be seen as a feature at the vehicle level. Hence, as such it can be modeled using feature diagrams as shown in Figure 34.

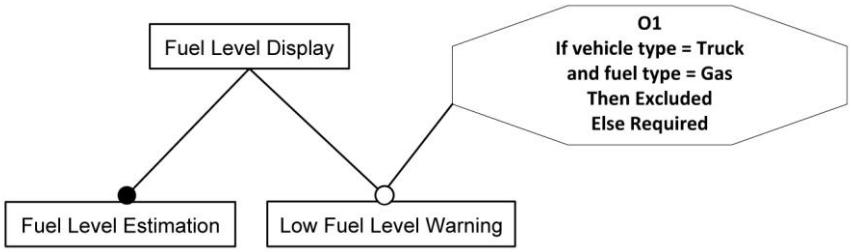

Figure 34: Feature diagram of the FLDPL's functionality

In the presented diagram, the Fuel Level Display is the main feature and can be decomposed into two sub-features: Fuel level estimation and Low fuel level warning. Since the first one is present in all the vehicles, it is linked with a mandatory relation. While, the Low Fuel Level Warning feature is optional, since it is not valid for trucks with gas fuel type. Hence, from a product line viewpoint it is an optional feature and it is represented in the diagram with an optional relation.
In accordance with the proposed extension for feature diagrams in Section 5.3, the condition on the selection of this optional feature is expressed in the obligation O1.

Hence, the diagram shown in Figure 34, will have a different configuration depending on the vehicle and fuel type. If a truck with gas fuel type is considered, then only the Fuel Level Estimation feature will be present. In all the other cases, both features will be included.

6.1.3 Product line behavior

For showing the behavior of the FLDPL, as described in Section 4.3, activity diagrams will be used. Furthermore, an approach for supporting product lines has been proposed in Section 5.4. The behavior of the different versions of the Fuel Level Display system has already been explained in Section 2.9. Instead, in this section the entire product line is considered. The product line behavior is depicted through the activity diagram shown in Figure 35.
As also described in the previous section, the main functionality of the systems composing the FLDPL are the Fuel level estimation and display and the Low fuel level warning. In order to perform these functions, the first operation that is carried out, is the reading of the fuel level in the fuel tank. For trucks with gas fuel type this value is directly presented to the driver and the low fuel level checking is not performed. Instead, for buses with gas fuel type, in addition to be shown to the driver, the read fuel level is used to determine if the fuel level is low. In case of trucks or buses with liquid fuel type instead, the read fuel level is filtered in order to reduce the noise due to movements of the fuel in the tank. For trucks with liquid fuel type a Kalman filter is applied. Instead, for buses a Low-Pass filter is used. After that the fuel level has been filtered, the low fuel level warning checking is performed similarly as for buses

with gas fuel type. At this point if the fuel level is below the minimum fuel level, a warning lamp in the instrument cluster is activated. Else, the warning lamp is deactivated. If the warning was already deactivated, then there is no effect on the systems. At this point no additional operations are performed.

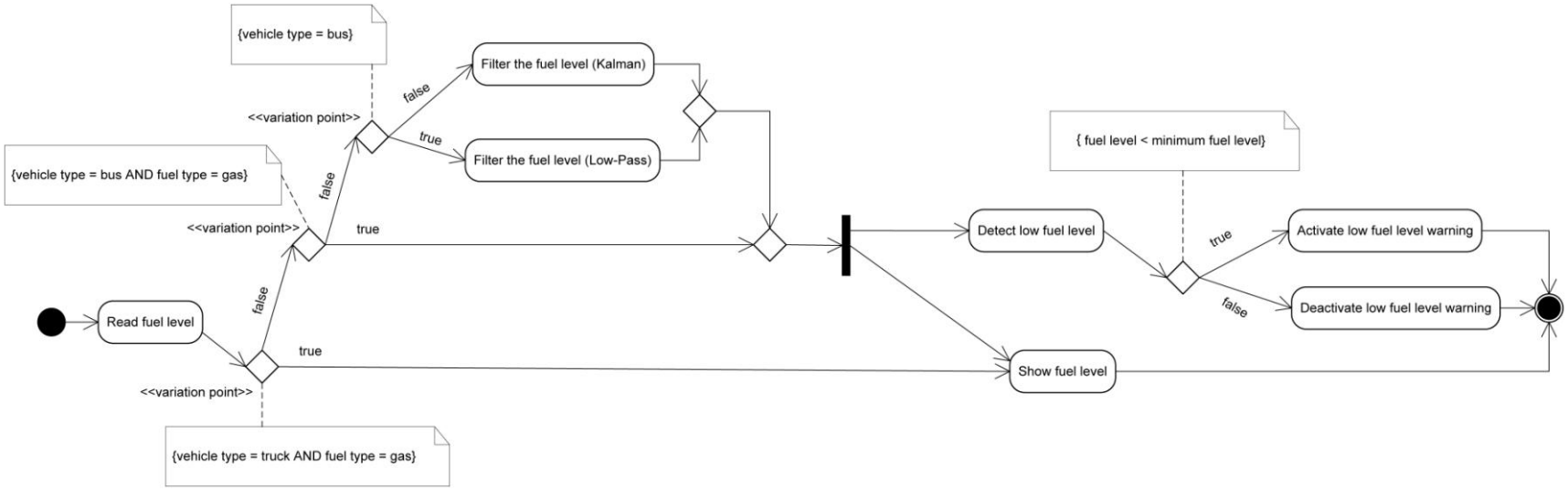

Figure 35: FLDPL behavior

6.1.4 Preliminary architecture

For describing the product line in terms of interfaces, interactions and elements of the items, in accordance with ISO 26262, a preliminary product line architecture can be created.
All the main elements composing the different versions of the Fuel Level Display System have already been described in Section 2.9. In that section each version of the Fuel Level Display System has been described separately. In this one instead, all the versions of the system will be treated as a unique product line, hence a product line approach is required for this purpose.
As described in Section 4.5 the same approach used in EAST-ADL has been chosen for this thesis work, in order to describe the architecture of the product line. Using SysML block definition and internal block diagrams, it is possible to represent a product line from an architectural viewpoint.
The work, which has been done for this purpose, was to merge the single preliminary architecture for each single considered version of the Fuel Level Display System, in order to describe them from a product line viewpoint. The result of this activity is shown in Figure 36.

In the presented preliminary architecture all the elements composing the FLDPL are considered. All the ECUs, sensors and other devices (fuel gauge and low fuel level warning) are present in the same architecture, since now the entire product line is considered. In addition since it describes different versions of the Fuel Level Display System, several variation point are present.

As Figure 36 shows, there are three different variation points. One related to the EMS, one to the COO and another one to the ICL. For each variation point, all the possible variants are described. For example we can see that for the variation point related to the EMS there are two possible alternatives. For the one related to the COO there are four alternatives and for the one related to the ICL there are three possible choices.
The selection of the variants it is not done arbitrarily. As already mentioned, it is done based on the feature selected in the usage context.

In order to increase the understandability of this architecture, instead of using one single feature diagram for showing the mapping between the variants, for each variation point, and the relative features, four copies of the same feature diagram have been used. This is resulted in a more understandable figure, since for each variation point there is a different feature diagram, for showing the relation between the variants and the features.
Concerning the variation point related to the COO, since there are four possible variants, two feature diagrams have been used. This is, because with one feature diagram the mapping was not clear. Furthermore, if for one variant there is more than one feature associated, it means that all the associated feature have to be selected in order to select that variant.

From this preliminary architecture, it is possible to derive all the possible allowed configurations representing the Fuel Level Display System. Where, the allowed configurations are derived based on the usage context.

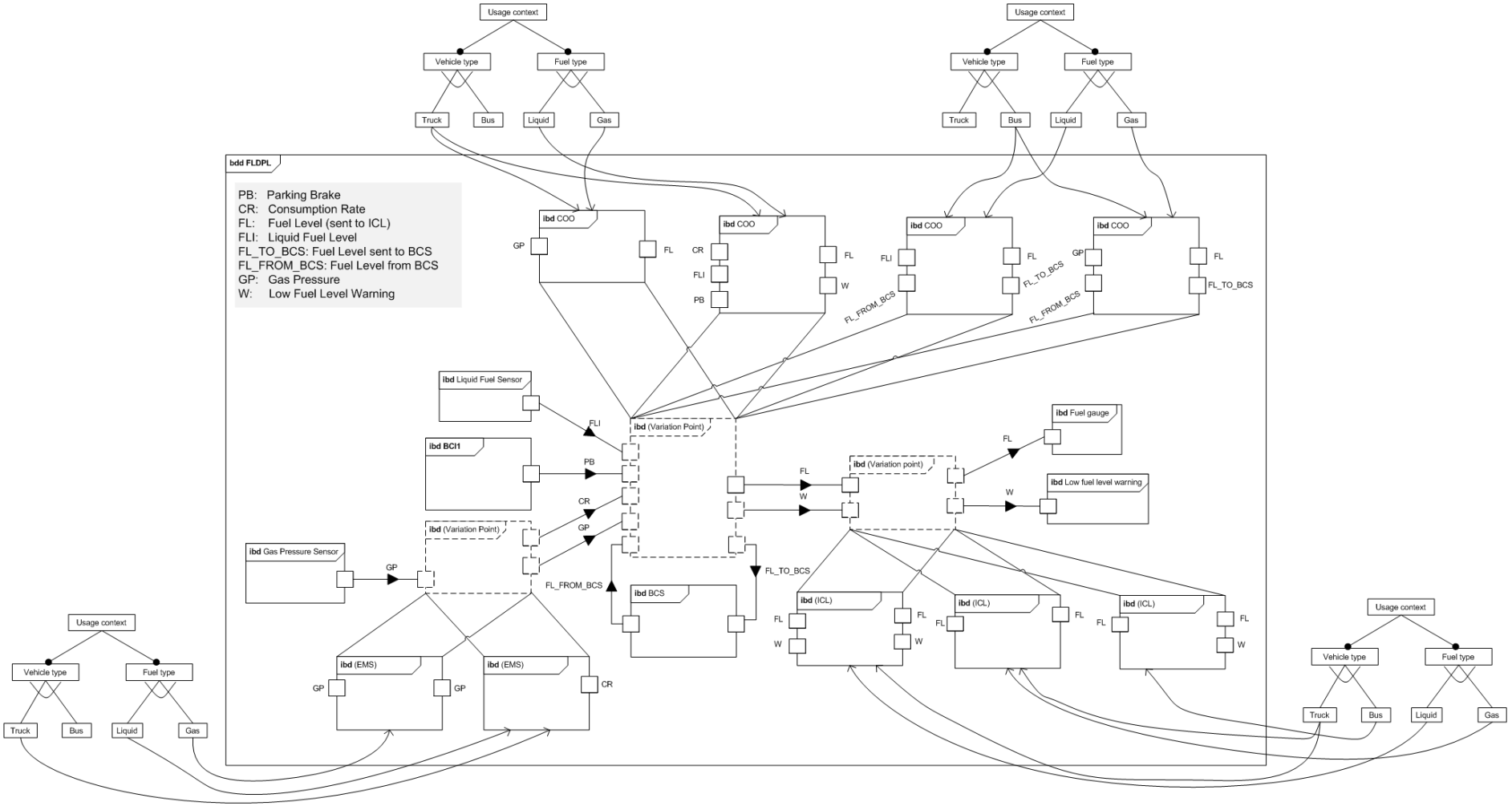

Figure 36: FLDPL preliminary architecture

For example if we consider a truck with liquid fuel type, a specific configuration of the presented architecture can be derived. Concerning the variation point related to the EMS, we can see that one variant is related to the feature *Gas,* and another one is related with the features *Truck* and *Liquid.* Since we are just considering a truck with liquid fuel type, the relative variant can be selected and the other one is excluded. Since this variant has not the interface for the *Gas Pressure Sensor (GP),* it means that it will be not present in this product line configuration. Instead, concerning the variation point related to the COO, there are four possible configurations. But, looking at the mapping with the usage context features, it is possible to see that to one variant corresponds the features *Truck* and *Liquid.* Meaning that it is the one that is selected for this case. In the selected variant, it is possible to see that there are not the interfaces that connect the COO to the BCS(FL_TO_BCS and FL_FROM_BCS). This means that the BCS will be not present in the product line configuration. Finally, for the variation point related to the ICL, there are three possible choices. As already done for the other variation points, it is possible to see that there is a variant mapped with the *Truck* and *Liquid* features. Meaning, that it has to be chosen for this product line configuration.

Hence, at this point it is possible to say that, for a truck with liquid fuel type the ECUs that are present in the product line configuration are the EMS, BCI1, COO and ICL. In more detail, looking at the messages flow and to the description provided in Section 2.9 it is possible to derive the system behavior. That is, the EMS calculates the fuel consumption rate and sends it to the COO. The BCI1 sends the parking brake status to the COO. The COO reads the Fuel level through the Fuel Level Sensor. With the read fuel level and the information received from the EMS and BCI1, the COO estimate the fuel level. In addition, it sets to true the low fuel level warning if the fuel level is low, else to false. At this point the COO sends the estimated fuel level and the low fuel level warning to the ICL. The ICL shows the fuel level to the driver through the Fuel gauge and activateS the low fuel level warning if the relative parameter (W) is true.

In the same way, it is possible to derive the configuration of the presented product line architecture for all the combination of the features, in the usage context feature diagram.

6.1.5 Functional requirements

For each version of the Fuel Level Display System presented in Section 2.9, a wide set of requirements was available in Scania. Due to time constraints, only a subset of them has been considered. This is, in order to highlight the concepts behind the approach that has been proposed to model this aspect of the product line, instead of getting lost in the wide set of requirements.
Since the requirements were not organized using a product line approach, also a commonality and variability analysis among the different version of the Fuel Level Display System has been performed.

Once defined the subset of requirements to consider, and identified which were the common and which the variable ones, they have been represented using a product line approach. In accordance with the approach proposed in Section 4.2 and extended in Section 5.3 the requirements have been modeled using feature diagrams. The feature diagram resulting

from the performed commonality and variability analysis is shown in Figure 37.

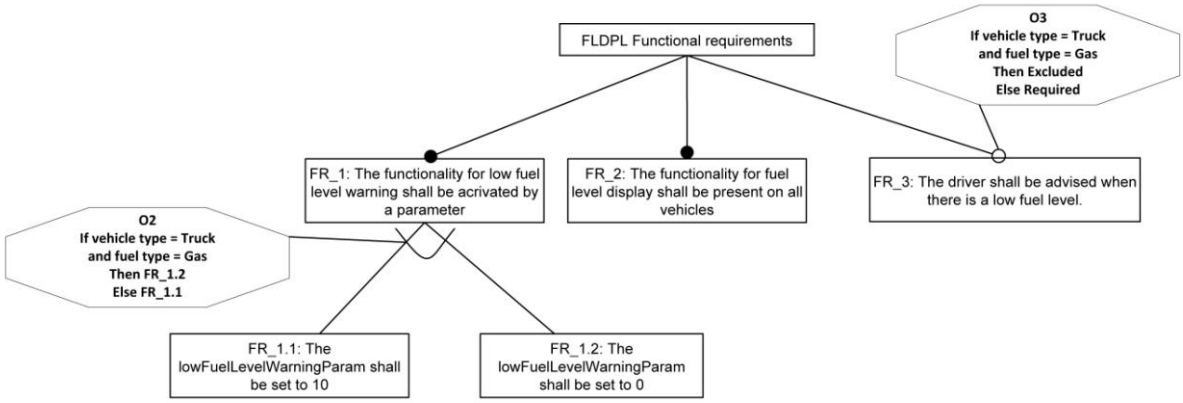

Figure 37: Feature diagram of the FLDPL's functional requirements

In the feature diagram depicted in Figure 37, the functional requirements for the FLDPL are presented. Five requirements have been considered. FR_1 and FR_2 are mandatory requirements. This means that they are valid for all the configurations of the product line. FR_1 is decomposed in two requirements, FR_1.1 and FR_1.2. Between these requirements there is an alternative relation. This means that only one of them can be selected in a product line configuration. The condition on their selection is described in the obligation O2. It is possible to see, from this obligation, that when a truck with gas fuel type is considered, FR_1.2 is required. In all the other cases FR_1.2 is excluded and FR_1.1 is required.

Concerning FR_3, it is possible to see that it is an optional requirement. This means that in some configurations of the product line it will be present and in other ones it will be not. Its presence is defined based on the condition presented in the obligation O3. It is possible to see that when a truck with gas fuel type is considered, then this requirement is excluded. Else, it is included in the product line configuration. The condition described in the obligation O3, is the same as the obligation O1 in Figure 34. This is, because the requirement FR_1.3 is relative to the Low Fuel Level Warning functionality. This functionality is not present in trucks with gas fuel type. Hence, it is logic that also the relative requirements are not present in the product line configuration.

The condition in the obligation O3 could also be expressed in relation with the feature diagram in Figure 34. That is, when the Low Fuel Level Warning functionality is excluded then also FR_1.3 is excluded from the product line configuration. However, the result is exactly the same.

6.2 Hazard analysis and risk assessment

As mentioned in Section 4.4.1 the HAZard and OPerability study (HAZOP) technique has been chosen as hybrid hazard analysis technique for this thesis work. It has already been extended for complying with ISO 26262, and in Section 5.5 an approach to adapt this technique to product lines has been proposed. In this section, the proposed adapted HAZOP technique is performed on the FLPDL. Table 9 depicts the adapted HAZOP table resulting

from the performed hazard analysis.

ID	Condition	Item	Attribute	Guide word	Operational situation	Deviation	Consequences	S	E	C	ASIL	Possible Causes	Variation Points
H1	**1- If fuel type = Gas Then V1 ignored** **2- If Vehicle type = Truck Then V1.Filter = Kalman Else V1.Filter = Low-Pass**	Fuel level estimation	Value	Higher	Highway road, good visibility condition, normal traffic, bad road conditions (i.e. snow), speed >40km/h	The indicated fuel level is higher than the actual fuel level.	Unpredictable luck of fuel. Leading to severe injuries (due to e.g. front/rear collision with other vehicles)	S3	E3	C3	C	1- Electrical or mechanical error in fuel level sensor 2- **{V1}** filter failure	**V1: Type = {Alt}, Filter = {Kalman, Low-Pass}**
H2	**1- If fuel type = Gas Then V2 ignored** **2- If Vehicle type = Truck Then V2.Filter = Kalman Else V2.Filter = Low-Pass**	Fuel level estimation	Value	Lower	Highway road, good visibility condition, normal traffic, bad road conditions (i.e. snow), speed >40km/h	The indicated fuel level is lower than the actual fuel level.	Annoyed driver. Trust issues for the Fuel Level Display	S0	E3	C0	-	1- Electrical or mechanical error in fuel level sensor 2- **{V2}** filter failure	**V2: Type = {Alt}, Filter = {Kalman, Low-Pass}**
H3	**1- If vehicle type = Truck AND fuel type = Gas Then H3 not valid**	Low fuel level warning	Occurrence	Omission	Highway road, good visibility condition, normal traffic, bad road conditions (i.e. snow), speed >40km/h	The low fuel level warning hasn't been activated when the actual level is low	Unpredictable luck of fuel. Leading to severe injuries (due to e.g. front/rear collision with other vehicles)	S3	E3	C3	C	1- Faulty warning lamp 2- CAN bus failure	
H4	**1- If vehicle type = Truck AND fuel type = Gas Then H4 not valid**	Low fuel level warning	Occurrence	Commission	Highway road, good visibility condition, normal traffic, bad road conditions (i.e. snow), speed >40km/h	The low fuel level warning has been activated when the actual level is not low	Annoyed driver. Trust issues for fuel level warnings.	S0	E3	C0	-	1- Faulty warning lamp 2- CAN bus failure	

Table 9: Adapted HAZOP table for FLDPL

From Table 9 it is possible to see that there are basically four deviations, which are:
1- The indicated fuel level is higher than the actual fuel level
2- The indicated fuel level is lower than the actual fuel level
3- The low fuel level hasn't been activated when the actual level is low
4- The low fuel level has been activated when the actual level is not low

These deviations have been considered with a specific operational condition, which is when the driver is driving the truck on highway roads, with good visibility conditions, normal traffic. But, with bad road conditions due to snow on the road, that is common during Swedish winters, with a speed above 40 km/h.
Based on the considered operational condition the first and third deviations are the ones that could lead to an harm. While, the second and fourth ones only get the driver annoyed and cause trust issues in the relative function.
Consulting Table 1, Table 2 and Table 3 presented in Section 2.2.1, the severity, controllability and exposure has been assigned to the relative entry in the adapted HAZOP table. Furthermore, based on these parameters and Table 4 an ASIL has also been assigned to each deviation. For the deviations leading to an hazardous situation, and probably leading to severe injuries, an ASIL C has been assigned. While, for the other deviations the combination of severity, exposure and controllability based on Table 4, resulted in an undefined ASIL, meaning that no additional work is needed to avoid that deviations.

From the *Condition* column it is possible to see that the variation points V1 and V2 are not valid (ignored) when there is a gas fuel type. This means that the relative cause in the *Possible causes* field is ignored for vehicles with gas fuel type. This is, because H1 and H2 are relative to the Fuel Level Estimation function. That in case of gas fuel type, it is not performed, but the read fuel value is directly displayed to the driver. Hence, the cause of the deviation can only be a mechanical or electrical fault in the fuel sensor in case of vehicles with gas fuel type. The cause cannot be a fault relative to the estimation (filtering) function.
Instead, H3 and H4 are not valid at all, for trucks with gas fuel type. H3 and H4 are relative to

the Low Fuel Level warning function that in case of trucks with gas fuel type it is not present. Hence, it makes no sense to consider these deviations in the product line configuration.

Concerning H1, from the *Condition* column it is also possible to see, that when trucks with liquid fuel type are considered, V1 is instantiated with the variant *Kalman*. While, in the other cases (buses with liquid fuel type) the variant *Low-Pass* is chosen.
This is because in case of trucks with liquid fuel type a Kalman filter is used to reduce the noise in the read fuel level. Hence the deviation could be caused by a failure related to this type of filtering. In the same way, when buses with liquid fuel type are considered, since a Low-Pass filter is used, the deviation could be caused by a failure in the Low-Pass filtering.
The second variation point, V2 is relative to H2. This deviation is exactly the opposite as the one considered for H1. However, the instantiation of the variation point is done in the same way as for V1.

6.2.1 Safety goals

At this point, after that the hazard analysis has been performed and the hazards have been identified, safety goals must be defined, in order to address these hazards as already described in Section 2.2.1.

Since the set of versions of the Fuel Level Display System have not been developed in compliance with ISO 26262, there were no explicit safety goals. Only functional requirements have been defined, and some of them were implicitly safety goals. Hence, after a study of these requirements, two of them have been selected as safety goals, addressing the identified hazards. In accordance with Section 4.2 they have been represented through feature diagrams. And, as explained in Section 5.3, obligations have been used to describe the conditions on the variability selection, within the feature diagram. The feature diagram representing the safety goals for the FLDPL is shown in Figure 38.

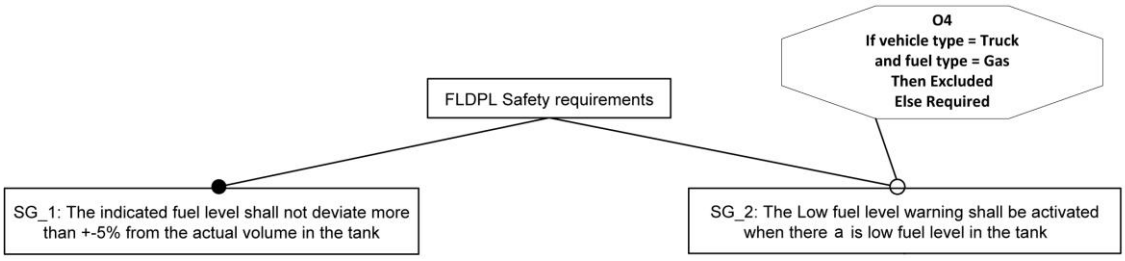

Figure 38: FLDPL safety goals feature diagram

From the presented feature diagram it is possible to see that for the FLPDL there are two safety goals, SG_1 and SG_2. SG_1 is a mandatory requirement, meaning that it is present in all the product line configurations. It has been defined to mitigate H1 in Table 9. It states that the fuel level shall not deviate more than +-5% from the actual volume in the tank. This is to avoid that an higher fuel level is shown to the driver, avoiding an hazardous situation.
Instead, SG_2 is an optional requirement. From the obligation O4, it is possible to see that it is excluded in case of trucks with gas fuel type. In all the other cases it is included in the

product line configuration. This is because for trucks with gas fuel type, there is not the low fuel level warning function. Hence, there is no need to define a requirement for an hazard that cannot never happen for this type of vehicles. SG_2 has been defined in order to mitigate H3 in Table 9, since it states that the low fuel level warning shall be activated when there is a low fuel level in the tank. Avoiding in this way that the low fuel level warning doesn't activate when there is a low fuel level in the tank, and avoiding an hazardous situation.
These safety goals inherit the ASIL of the relative hazard. Hence both of them have an ASIL C.

6.3 Functional safety concept

Once that safety goals have been defined, they have to be decomposed in functional safety requirements, as already explained in Section 2.2.1.
For each safety goals at least one functional safety requirement has to be defined. And there is need to keep the traceability between safety goals and the relative safety requirement(s). For this purpose, as described in Section 4.2 feature diagrams can be used to represent, in this case, functional safety requirements, and keeping the traceability between the relative safety goal.
In order to decompose the safety goals into functional safety requirements, an analysis of the available requirements for the set of versions of the Fuel Level Display System has been performed. This is, because in Scania, traceability among the considered requirements has not been found. Once identified the requirements decomposing the safety goals, the feature diagram shown in Figure 38, has been extended, by adding functional safety requirements as child nodes for each safety goal. The extended feature diagram is shown in Figure 39.

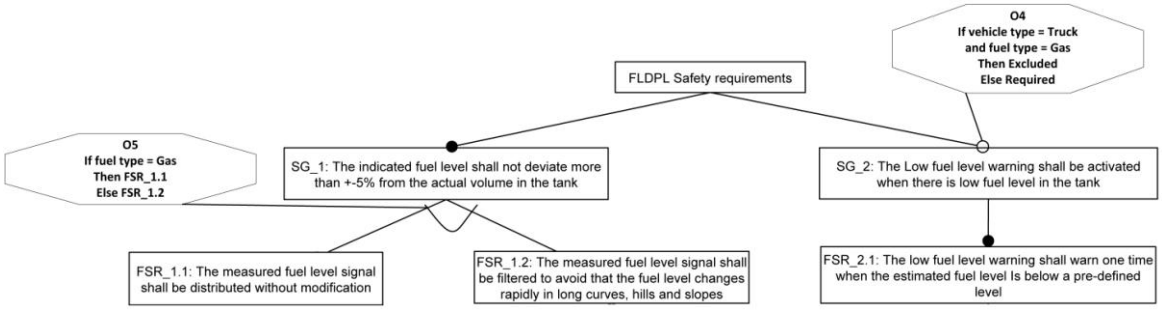

Figure 39: FLDPL safety goals and functional safety requirements

In the extended feature diagram, the safety goals and the relative functional safety requirement(s) for the FLDPL are presented. It is possible to see that SG_1 is decomposed into two functional safety requirements, which are connected with an alternative relation. The obligation O5 describes the condition on their selection. If a vehicle with gas fuel type is considered, then FSR_1.1 is selected, else FSR_1.2. This is because for vehicles with gas fuel type the fuel level read from the fuel tank doesn't have to be filtered. Hence it has to be transmitted (distributed) without modifications. Instead, for vehicles with liquid fuel type, the read fuel level has to be filtered, in order to avoid that the fuel level changes rapidly in long curves, hills and slopes.
SG_2 instead, is decomposed in one functional safety requirement, which is FSR_2.1. It states that the low fuel level warning shall warn one time when the estimated fuel level is below a pre-defined level. In this way, when the fuel level is below the minimum amount of

fuel, the driver is advised. SG_2 is decomposed in the same way for all the product line configuration, in which it is present.

6.4 Technical safety requirements

Once that safety goals have been decomposed in functional safety requirements, another step required from ISO 26262, is to decompose functional safety requirements in technical safety requirements, as described in Section 2.2.1. Similarly as done for safety goals and functional safety requirements, the requirements decomposing the functional safety requirements has been identified among the ones available in Scania. Then, they have been organized in a feature diagram, presented in Figure 40, extending the one shown in Figure 39.

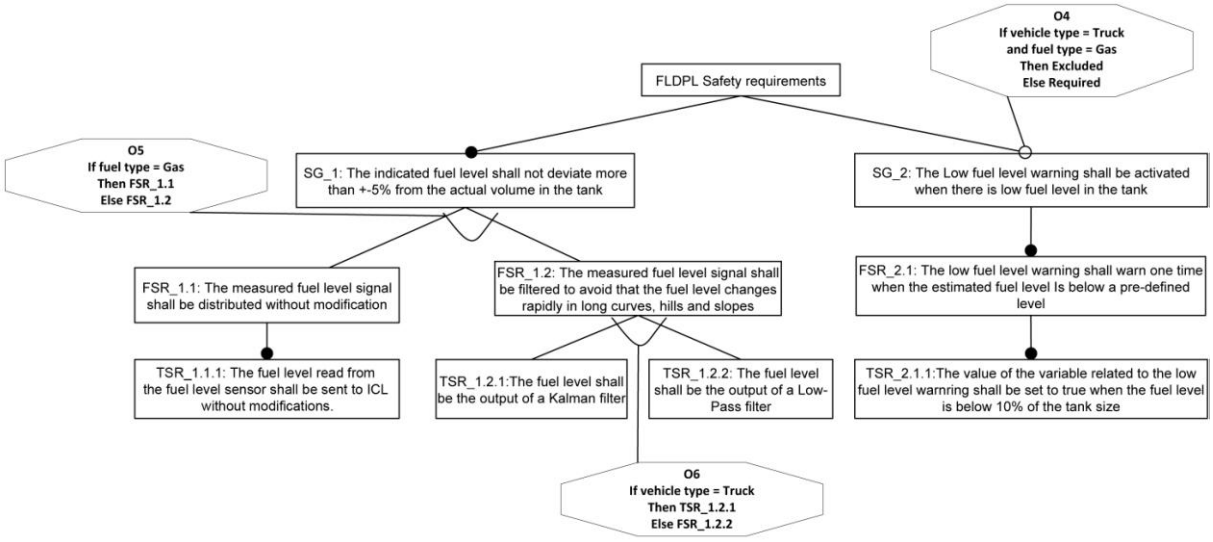

Figure 40: FLDPL safety requirements for the FLDPL

As the feature diagram in Figure 40 shows, each functional safety requirement is decomposed in at least one technical safety requirement. FRS_1.1 is decomposed in TSR_1.1.1. This safety requirement states that the fuel level read from the fuel tank shall be sent to ICL without modifications. This is in order to keep the same value as the one read in the fuel tank through the gas pressure sensor. FSR_1.2 instead, is decomposed into two technical safety requirements, TSR_1.2.1 and TSR_1.2.2. The obligation O6, describe which one to choose. In case of trucks, TSR_1.2.1 is chosen. This is because in trucks with liquid fuel type the fuel level is filtered using a Kalman filter. Instead, in case of buses FSR_1.2 is decomposed in TSR_1.2.2. This safety requirement states that the fuel level shall be filtered using a Low-Pass filter.
Finally FSR_2.1 is decomposed in TSR_2.1.1. This requirement states that the fuel level warning shall be activated when the fuel level is below the 10% of the tank size. In this way the driver has the time to refill the fuel tank, avoiding in this way hazardous situations.

6.5 System design

In Section 6.1.4 the preliminary product line architecture has been presented. In this section instead, the product line architecture will be described in more detail. Ideally, each ECU that has been considered in the architecture shown in Figure 36 should be examined and described in more detail. Most of the information gathered, during the thesis work in Scania, was related to the COO and ICL, while for the other ECUs the information was lacking. For this reason, and also because these ECUs are the most significant ones, it has been decided to focus only on the COO and the ICL. Hence, both ECUs will be considered in this section.
As already done for the preliminary architecture in section 6.1.4, the structure of the ECUs will be described using block definition and internal block diagrams, in accordance with Section 4.5. In this case, the block diagrams are used to represent the entire ECUs, while the internal block definition diagrams are used to represent the components within the ECUs.

The first examined ECU is the COO. The architecture of this ECU is shown in Figure 41. The presented architecture describes the internal structure of the COO, from a product line viewpoint. It is possible to see that there are three variation points. Starting from the top of Figure 41, the first one is related to the type of filtering used to filter the fuel level read from the fuel tank. For this variation point there are two possible variants. One where a Kalman filter is used and another one where a Low-Pass filter is used instead. When the variant with a Kalman filter is used, the information related to the parking brake status (PB) and the engine consumption rate (CR) are used to estimate the actual fuel level. If the parking brake is on, meaning that the vehicle is parked, the Kalman filter is not applied. After that the fuel level has been filtered, the value is sent to the LowFuelLevelWarning component, in order to determine if the fuel level is low. In case of a low fuel level the low fuel level warning parameter (W) is set to true. Then, the low fuel level warning parameter and the filtered fuel level (FL) are sent to the ICL. This variant is selected in case of trucks with liquid fuel type.
Instead, when the other variant is selected a simple Low-Pass filter is used on the read fuel level (FLI). After that the fuel level has been filtered the updated value (FL_TO_BCS) is sent to the BCS, without applying the control for a low fuel level. Then, as it will be also explained later, the BCS forwards back this value to the COO. This variant is selected in case of buses with liquid fuel type. For this type of vehicles the low fuel level control is performed directly within the ICL.
In case of vehicles with gas fuel type, the variation point is not instantiated, since the fuel level filtering is not performed. Meaning that it will be ignored, and the relative variants will be not present in the product line configuration.

The second variation point instead, is related to vehicles with gas fuel type. If a bus with gas fuel type is considered, then the gas pressure (GP) is sent to the BCS. Instead, if a truck with gas fuel type is considered, the (GP) is sent directly to the ICL. Hence, in case of buses with gas fuel type the variant *To BCS* is selected. While, in case of trucks with gas fuel type the variant *To ICL* is chosen. In case of vehicles with liquid fuel type, this variation point is ignored. Since the gas pressure is not relevant for those vehicles.

Finally, the third variation point is relative to the messages received from the BCS. In case of buses the COO sends the filtered fuel value to the BCS. Then the BCS sends it back to the COO. At this point the COO forwards it to the ICL. This variation point is considered only in case of buses. This is because the BCS is present only in those vehicles and not in trucks. In

case of trucks as explained above, the fuel level is sent directly to the ICL.

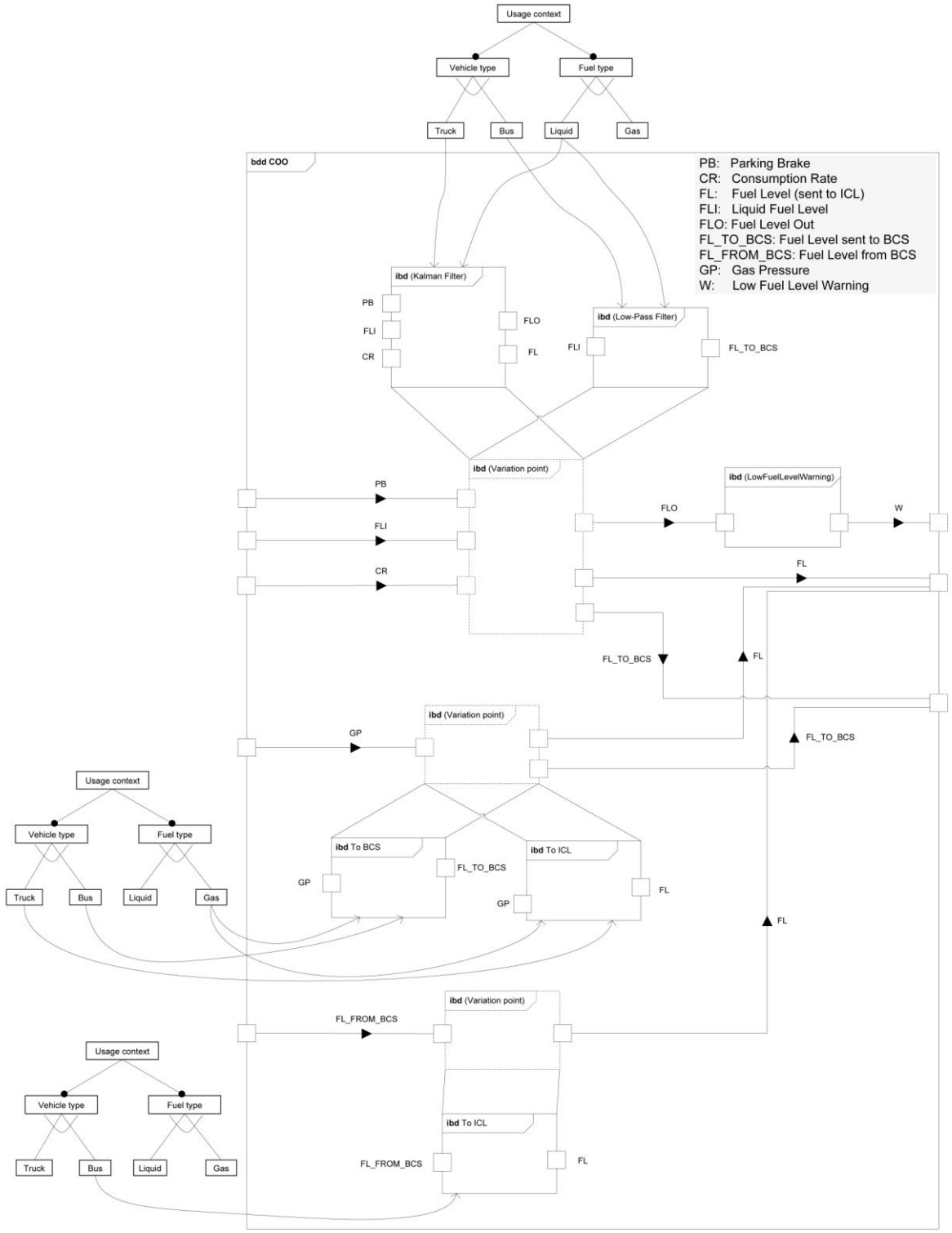

Figure 41: COO architecture

After that the COO has been described, the ICL is examined. The architecture representing the ICL is shown in Figure 42. This ECU is responsible for activating the low fuel level warning and setting the fuel gauge. Furthermore, as for the COO, there are different

configurations for this ECU.

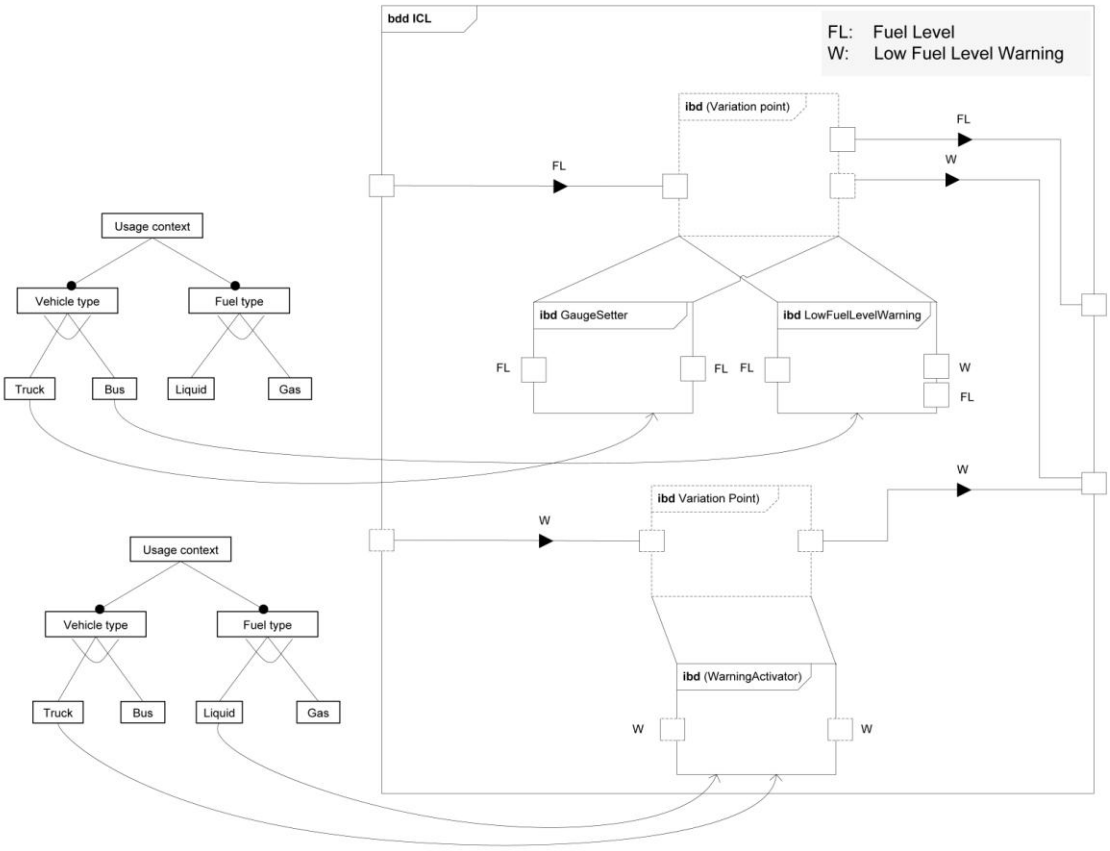

Figure 42: ICL architecture

From the architecture depicted in Figure 42, it is possible to see that there are two variation points. Starting from the top of the figure, the first variation point has two variants. In case of trucks with liquid fuel type, the fuel level signal (FL) is simply used, without modifications, to set the fuel gauge. Hence, in this case the variant *GaugeSetter* is chosen.
Instead, in case of buses the low fuel level warning control is done within the ICL. And, the fuel level is used to determine if the fuel level is low. Hence, the variant *LowFuelLevelWarning* is selected in case of buses.

The second variation point has only one variant, which is selected in case of trucks with liquid fuel type. In these vehicles the low fuel level warning is determined within the COO. Hence, the relative message is sent from the COO to the ICL, and it is directly used to activate/deactivate the low fuel level warning lamp. This variation point is considered only for trucks with liquid fuel type, because in the other cases the low fuel level warning message (W) is never received.

6.5.1 Safety analysis on system design

As explained in Section 2.2.2, another activity required in the ISO 26262 is the safety analysis on system design, in order to identify the causes of systematic failures and the

effects of systematic faults.
As described in Section 4.4.2 a deductive hazard analysis technique has been chosen for this purpose, the Fault Tree Analysis. In addition, it has been slightly modified as explained in Section 5.6. In this section the proposed Fault tree analysis is performed on the FLDPL.
Concerning the FLDPL, the Fault tree analysis has been performed taking also into account the HAZOP table presented in Table 9. And the fault tree deriving from the performed safety analysis is shown in Figure 43.

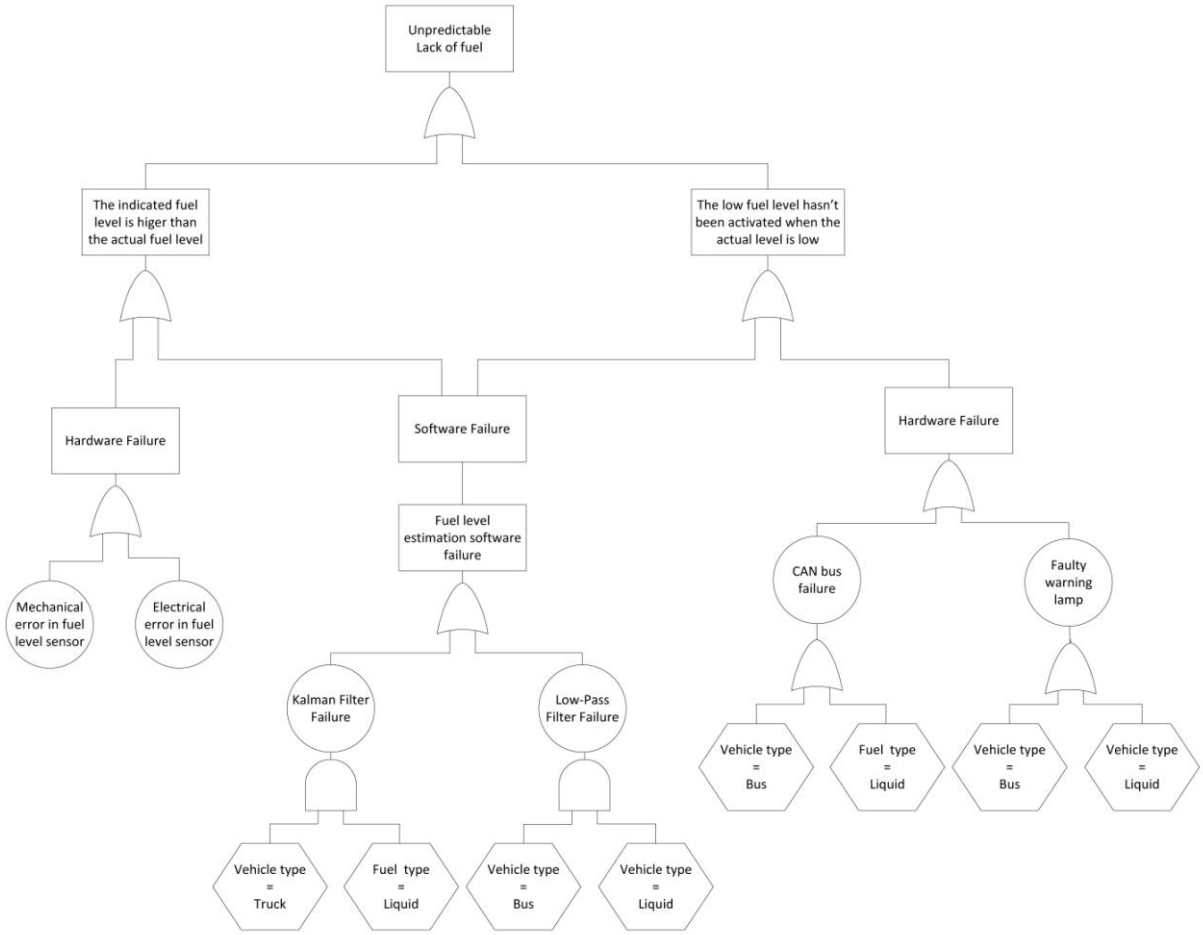

Figure 43: Fault tree analysis for FLDPL

The top event of the presented fault tree, is "Unpredictable lack of fuel". It occurs when the fuel level presented to the driver is higher than the actual fuel level, or if the low fuel level warning light is not activated when supposed to.

When the fuel level presented to the driver is higher than the actual one, it can depend on hardware or software causes. It is possible that there can be a mechanical or electrical error in the fuel level sensor and the wrong fuel level is transmitted to the relative ECU. Hence, if one of these errors occur, the fuel level shown to the driver can be higher than the actual fuel level.
The top event can also be caused by a software failure in the fuel level estimation software. This failure can be caused by a failure related to the Kalman filter or a failure related to the Low-Pass filter. The type of failure depends on the vehicle type and fuel type. As it is

possible to see from the presented fault tree for the *Kalman filter failure* there are two conditions related with an *AND* gate. This means that both have to be verified in order to include the relative event in the product line configuration. One condition is related to the type of vehicle and another one to the fuel type. It is possible to see, that only when trucks with liquid fuel type are considered, the *Kalman filter failure* event in included in this fault tree. In the same way, the *Low-Pass filter failure* event is included in the fault tree only when buses with liquid fuel type are considered. When vehicles with gas fuel engine are considered instead, a failure in the estimation software is excluded, since it cannot ever happen. This is because for vehicles with gas fuel type the fuel level read from the fuel tank is not filtered. But, it is sent directly to the ICL without modifications.

The other possible event causing the unpredictable lack of fuel is when the low fuel level warning is hasn't been activated when the actual fuel level is low. It can depend both on software and hardware causes. The software causes are exactly the same as the ones described when the fuel level presented to the driver is higher than the actual fuel level. Instead, the hardware failures can depend on a *CAN bas failure* or a *faulty warning lamp*. When there is a can bas failure, it is possible that the received low fuel level warning message is wrong and the relative lamp is not activated. While, if there is a faulty warning lamp, also when the fuel level warning message is received correctly the lamp is never activated. Both *CAN bus failure* and *faulty warning lamp* events are the causes considered for every product line configuration, except for trucks with gas fuel engine. This is, because in those vehicles the low fuel level warning functionality is not present and the causes with the relative event are not considered in the fault tree analysis. Hence, when a truck with gas fuel type is considered, it is not possible that the warning light is not activated when supposed to, since it is not present at all in the product line configuration. Hence, the relative part in fault tree will be cut, and not considered.

6.6 Safety case

After that all the necessary evidences have been collected, Following ISO 26262 the safety case is a collection of all the work products. However, simply collecting the work products is not enough to show the system acceptable safety. But there is need to structure them in a way that it is easier to understand in which way they contribute to the system safety. For this purpose, as explained in Section 4.6, the Goal Structuring Notation (GSN) is used to create the structure of the safety case for the FLDPL.

A safety case is composed from process-based evidences and product-based evidences (Section 2.4.1). Process-based evidences can be used to show that the activities required from ISO 26262 have been performed. While, product-based evidences can be used to show that the system behaves in a safe way. Hence, that all the hazards have been identified, and mitigated. Based on these classes of evidences, a first structure of the safety case could be defined by the one shown in Figure 44.

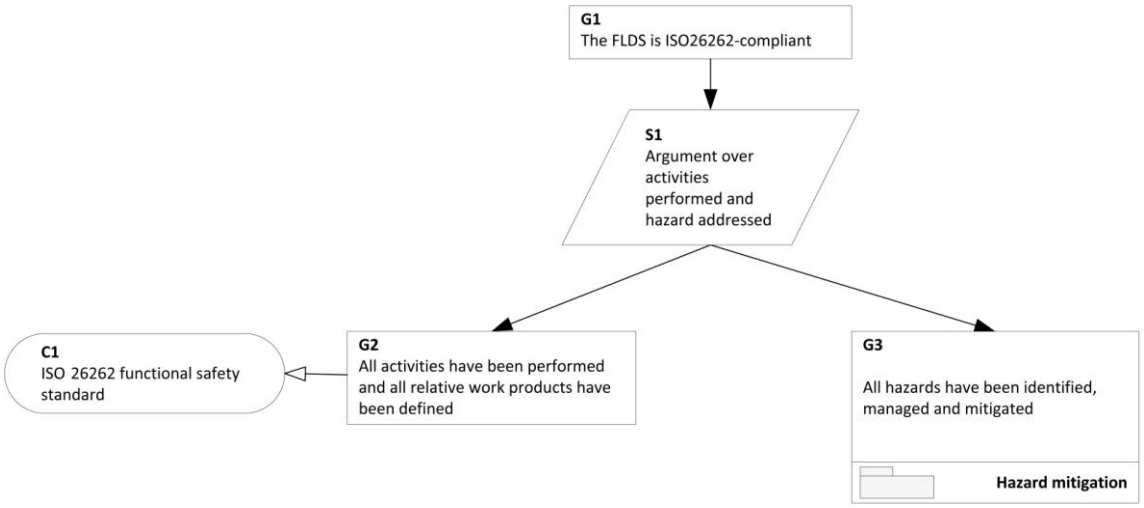

Figure 44: Top level goal structure of the safety case

This goal structure shows a clear division between process-based and product-based aspects. G2 is related to process-based aspects while G3 is related to the product based ones. Hence, showing how these goals are satisfied, imply that also the main goal is satisfied. That is, the compliance with ISO 26262, hence the acceptable safety of the product line.

In Section 6.6.1, it is shown how G2 is further decomposed. While the *Hazard mitigation module,* containing G3, is described in Section 6.6.2.

6.6.1 Process-based evidences

Showing that *all activities have been performed and all relative work products have been defined (G1)* in the context of ISO 26262*,* can be done showing the same thing for each section of the safety standard. Since for this thesis work, only Section 3 and 4 of ISO 26262 has been considered, the goal structure is limited to these sections. Hence, G2 can be developed as shown in Figure 45.

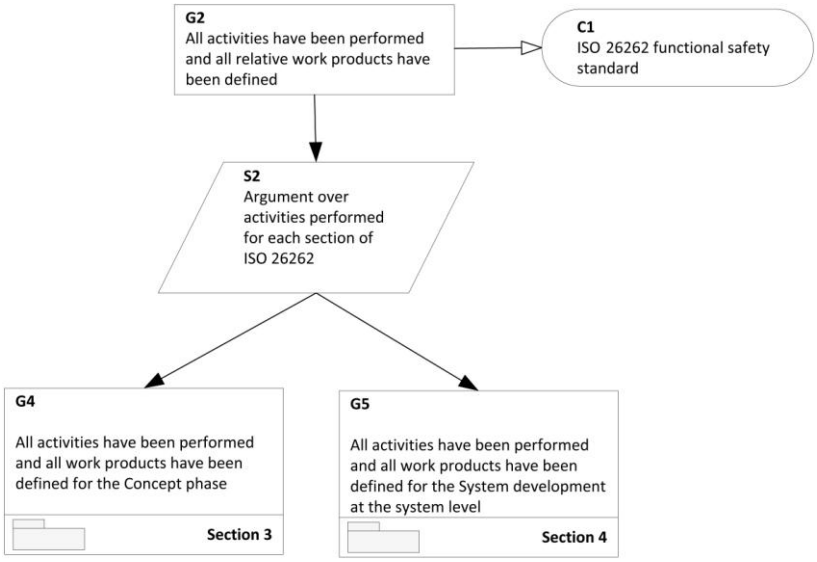

Figure 45: Process-based aspects

As Figure 45 shows, G2 is decomposed in two goals, G4 and G5. Each of them aims at showing that all the activities have been performed and all the relative work products have been defined, respectively for the Concept phase and the System development at the system level. Both G4 and G5, are away goals defined respectively in *Section 3* and *Section 4* modules.

Section 3 module is related to the Concept phase. Following ISO 26262, for this section, four activities with the relative work products, are required:

1. Item definition
2. Initiation of the safety lifecycle
3. Hazard analysis and risk assessment
4. Functional safety concept

Hence, to support G4 it could be stated that each of these activities has been performed. In addition, each claim could be directly supported from relative work product(s) as evidence. Since the scope of this thesis work is limited, only the evidences that have been developed are considered. Based on this, in Figure 46 it is shown how *G4*, within *Section 3* module, is decomposed.

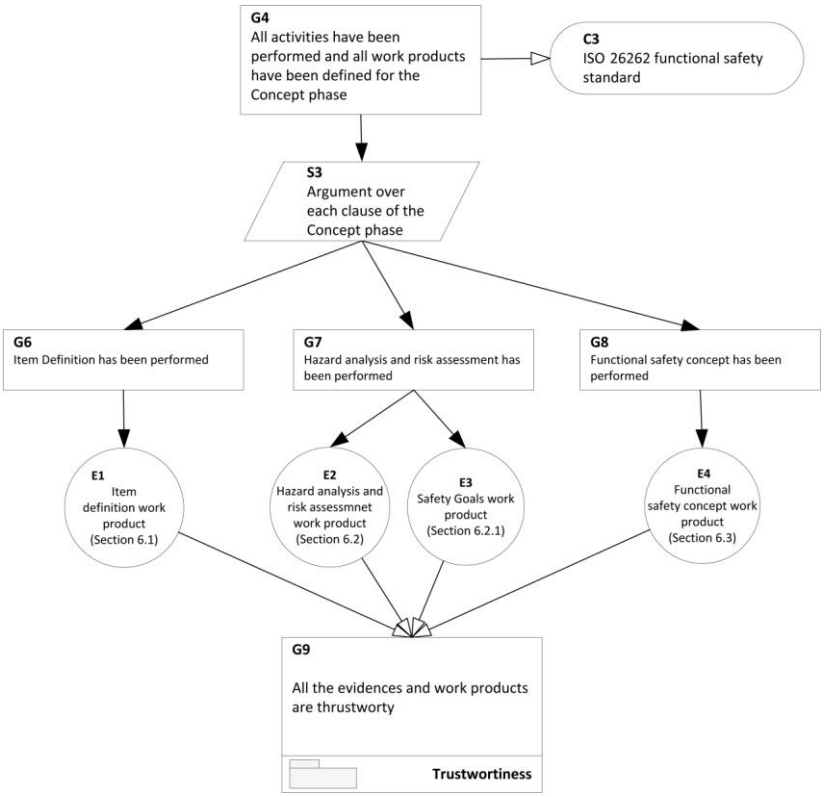

Figure 46: Section 3 module

Using a work product as evidence, for showing that an activity has been performed, is not enough to show the acceptable system safety. This is, because also the evidence itself has to be trustworthy. For this purpose the trustworthiness of the evidences is managed in the

Trustworthiness module.

Section 4 module instead, is related to the Product development at the system level Section of ISO 26262. The same strategy used for G4, can be used to develop the goal G5 within this module, but in the context of Section 4 of ISO 26262. The resulting goal structure is shown in Figure 47.

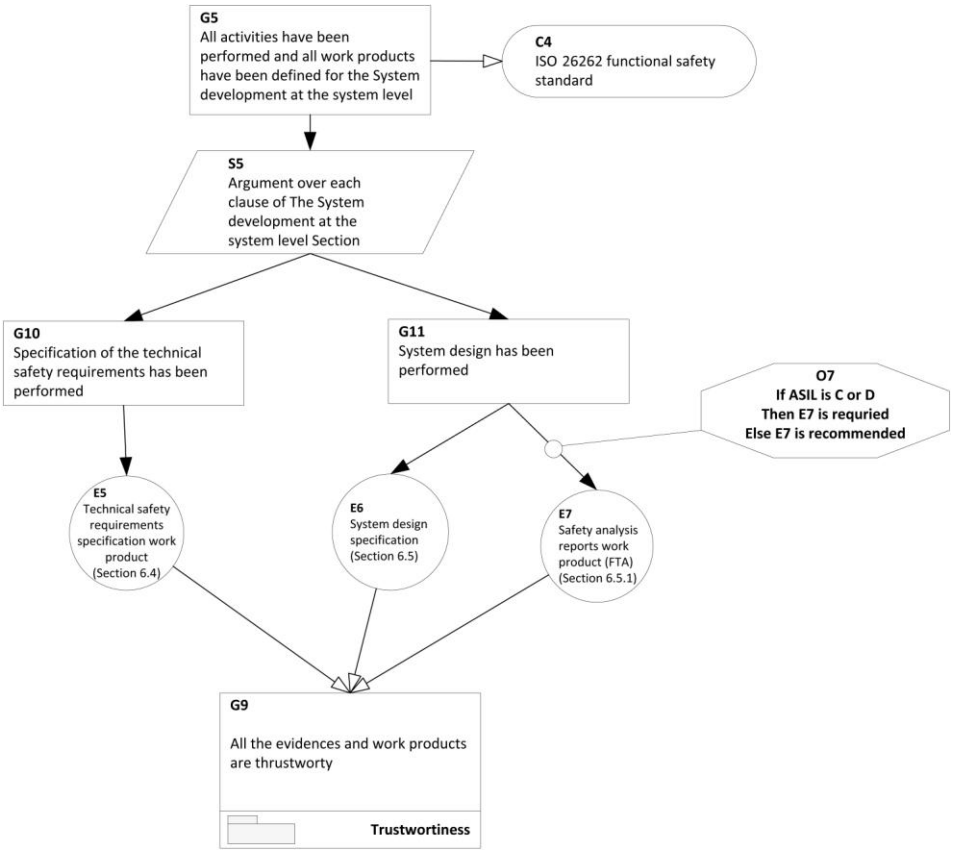

Figure 47: Section 4 module

To show how G5 is satisfied the clauses related to the fourth section of ISO 26262 are considered. In Figure 47 only the clauses that have been considered for this thesis work have been taken into account. *G10* and *G11* are related respectively to the technical safety requirements and system design. And, to support these goals, the work products required from ISO 26262 are used as evidences.
Furthermore, for systems having an ASIL lower than C the deductive hazard analysis (FTA) is not mandatory. Hence, the relative evidence (*E7*) is an optional one. Meaning that, it is only considered only for systems having an ASIL C or D, as described in the obligation *O7*.
In the same way as done for the goal structure presented in Figure 46, the trustworthiness of the evidences is supported within the *Trustworthiness* module.

The *Trustworthiness* module contains the goal structure showing the trustworthiness of the evidences, in the context of ISO 26262 and the Fuel Level Display System. As already mentioned, only providing an evidence in order to show the acceptable system safety is not enough. Since, its provenance can be uncertain. For this purpose the quality of the evidences has to be proven. The quality of the evidences can be argued, for example,

showing that the personnel that created the evidences is qualified; that the evidences are consistent with each other. This is, because since in this case a product line is the item, the evidences have to be configured in order to be valid for one single system. And this configuration has to be coherent within all the provided evidences; furthermore, the quality of the evidences can be also argued showing how specific tools have been developed or adopted in order to provide them.
This way of showing the trustworthiness of the evidences can be also described by a goal structure. This goal structure is shown in Figure 48 and it is exactly the one that is used within the *Trustworthiness* module.

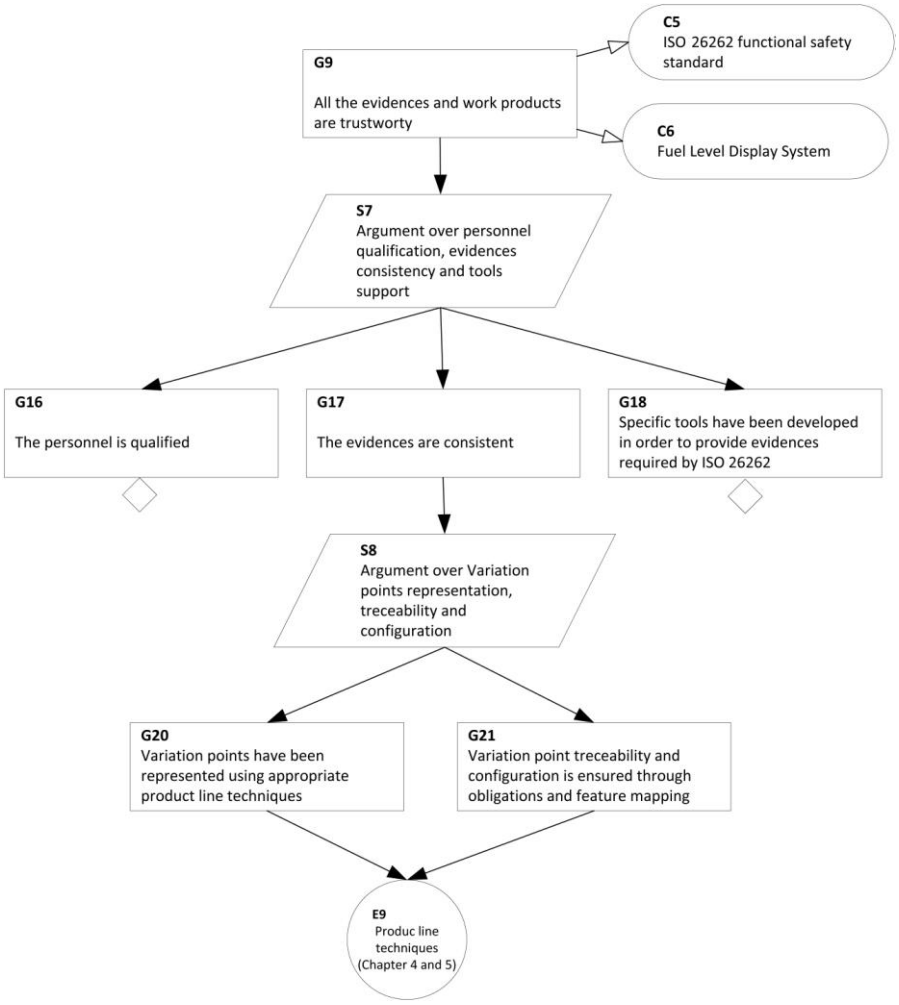

Figure 48: Trustworthiness module

In the depicted goal structure G9 is broken down in three goals (G16, G17 and G18) through the strategy S7. During this thesis work, there has not been the time for finding evidences regarding the qualification of the personnel (G16), and the specific tools adopted for generating evidences ISO-26262 compliant (G18). For this reason these goals has been marked as undeveloped.
Instead, the consistency of the evidences can be shown argued over the management of the variation points, within each provided evidence. In more detail, it can be shown the variation points have been represented using appropriate product line techniques (G20), and that their configuration and the relative traceability is ensured thanks to the obligations oriented

methods that have been used, and also thanks to the feature mapping for configuring the product line architecture (G21). To support these goals, the adopted product line techniques that has been used can be used as evidences.

6.6.2 Product-based evidences

Concerning product-based evidences, ISO 26262 doesn't state anything about how to show that the system behavior is safe. Hence, it has been decided to reach this goal showing that all hazards have been identified and mitigated, (G3). This is because if the hazards are correctly managed, the system behavior should be acceptably safe, in the context of the Fuel Level Display System. The proposed goal structure, related to product-based aspects, is shown in Figure 49.

All hazards have been identified during the Hazard analysis and risk assessment from a inductive and deductive (hybrid) viewpoint (using HAZOP analysis). Instead, the hazards have been analyzed from a deductive view point during the safety analysis on system design (FTA). Hence, the HAZOP table and Fault Tree can be used as evidences, for showing that all the possible hazards have been correctly identified (G12). Since, the FTA is mandatory only for systems having an ASIL C or D, it is available only in case of these ASILs. Hence, the FTA is an optional evidence and its usage for showing the safe behavior is related to the ASIL of the system. This is also described by the obligation O7. This obligation is exactly the same one within the goal structure shown in Figure 47.

For mitigating the identified hazards, safety requirements have been defined. These safety requirements include safety goals, functional and technical safety requirements. Hence, these requirements could be used as evidence showing the hazard mitigation.
But, the only definition of the safety requirements is not enough to show the safety behavior of the system. It has also to be shown that these requirements have been correctly implemented (G15).

During this thesis work no investigation has been done concerning the correctness of the implementation of the safety requirements, since it was out of the scope of the thesis. Hence, the relative goal (G15) has been marked as undeveloped.

As already shown and explained in the previous section, the trustworthiness of the evidences is argued in the Trustworthiness module.

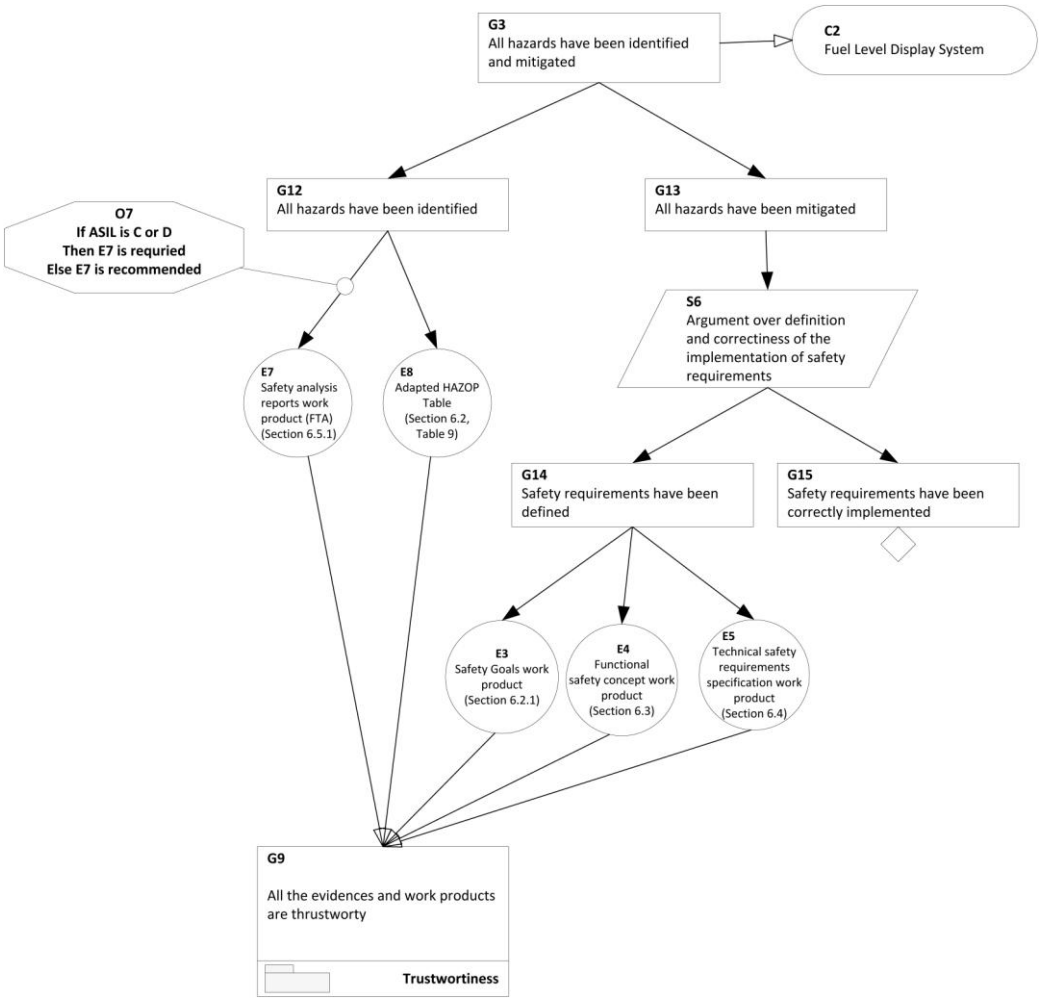

Figure 49: Hazard mitigation module

6.7 Discussion

In this chapter, it has been shown how to build a safety case for the FLDPL, with the chosen product line techniques. A goal structure concerning process-based arguments and another one concerning product-based arguments have been presented.

Concerning the goal structure relative to the process-based aspects, it doesn't contain many variation points, as it could be expected, since a product line is treated. There is only one, and it is related to the ASIL of the system. The lack of variation points is also due to the sections of ISO 26262 that have been treated during this thesis work, since they don't include many additional activities related to the ASIL of the system.
However the goal structure is very general and it is not strictly related to the structure of the product line. Instead, it is mostly based on the structure of the ISO 26262. This is because the decomposition of the goals follows the decomposition of the chapters in sections, and the sections in clauses. This is a real important aspect and thanks to this, the presented goal structure is valid for all the product line configurations, and potentially also for other product lines.
Furthermore, the lack of variation points within the goal structure doesn't mean that there aren't other ones within the safety case. Indeed, each evidence has been managed with

appropriate product line techniques, meaning that within each of them there are other variation points with the relative obligations. Hence, after that all the variation points within the evidences have been configured, based on a specific version of the system, they can be used to argue the acceptable system safety in accordance with the presented goal structure.

Concerning the *Trustworthiness* module, it is also not based on specific aspects of the product line, but it is very general. And, in the same way as for the other process-based evidences, it is valid for all the product line configurations.

Similarly for the goal structures related to process-based aspects, the one related to product-based aspects is also valid for the entire product line. This is because a general goal structure is used, combined with evidences that are managed using product line techniques. Hence, with the correct configuration of all the variation points within the used evidences, the safety case (Goal structure and evidences) are valid for the entire product line and all its configurations.

It has been noted, in this thesis work, that with the exception of the evidences used to show the provenance and trustworthiness of product-based and process-based evidences (evidences within the *Trustworthiness* module), the ones used to argue process-based and product-based aspects are exactly the same. Or in some cases, the evidences used for arguing product-based aspects are a part of the evidences used for arguing process based aspects. Furthermore, the evidences used in this thesis work are exactly the work products required by ISO 26262. This led to the conclusion that process-based and product-based evidences are mostly the same, but used from a different viewpoint.
Furthermore, in ISO 26262 the safety case is considered as a simple collection of work products, but many other definitions in literature define the safety case as a structured argument, composed by product-based and process-based evidences, showing the acceptable system safety. In this thesis work it has been found that the safety case is a collection of work products, but through a goal structure, structured in a way that is not difficult to understand their contribution to the system safety. In addition these work products, or a part of them, can be considered as process-based or product-based evidences, depending on the viewpoint from which they are used.

7. Conclusion

This conclusive chapter is organized as follows: In section 7.1 the summary of the outcome of the thesis work is presented: In section 7.2 directions for future works are described.

7.1 Summary

This thesis proposed an approach for building a safety case for safety critical product lines, in accordance with ISO 26262, the functional safety standard valid for the automotive domain. Furthermore an alignment of the product line development process with the safety activities required by ISO 26262 is proposed, since the safety standard does not contain guidelines for product lines. In order to validate the proposed approach a small-sized safety critical has

been used as case study.

In this thesis several product line techniques are accurately selected among the ones available in the considered literature, in order to develop a safety critical product line in accordance with ISO 26262. Such techniques give the possibility to specify, manage and trace commonalities and variabilities at each step of the development and safety life cycle defined in the safety standard. More specifically, the focus of this thesis work has been limited to the concept phase and part of the product development at the system level sections.

A key part of ISO 26262 is the building of a safety case for showing the acceptable safety of a system. In case of product lines, where numerous versions of the same system can be present, building a safety case can result in a very time consuming then costly activity. For this reason an optimization of the reuse of the safety argumentation is required. For this purpose a graphical notation, the Goal Structuring Notation, has been used for modeling the structure of the safety case, in order to clearly define the common and variable parts and optimize the reuse of the safety argumentation.

A case study of a fuel level display product line, developed by Scania, has been presented in order to show the validity of the proposed approach. Staring form the item definition until the system design, the selected product line techniques have been applied on the fuel level display system product line, following the system development and safety life cycle defined in ISO 26262. All the common and variable parts have been identified, and in addition for each variation point an obligation is defined in order to trace the configuration of the variation points within the entire product line work products, including in the safety case.

Limitations

Also if the goal of the thesis has been reached the current work presents some limitations:

- A limited part of ISO 26262 has been treated. Only the Concept phase and part of the Product development at system level sections have been took into account. In addition for each clause, some requirements have not been considered, since the information was not always available.

- The fuel level display product line shown only few functions that could be considered safety critical. Furthermore, all the members of the product line had the same ASIL, and it has not been possible to evaluate the impact of a feature on the ASIL of the product line, hence on the development and safety life cycle.

- The different versions of the fuel level display system have been entirely managed manually. There were no specific tools for managing the variation points within the different work products, meaning that the configuration of one single product have to be done manually. Without specific tools the proposed approach could be not feasible for larger product lines.

- The argument of this thesis was quite new within the company. Only a study on building a safety case for one version of the fuel level display system was available. In addition no thesis works that were considering a product line were available. Discussion and assessment from people that worked before on safety cases and product lines could probably improve the quality of the thesis work.

7.2 Future work

In order to be applied in the real case, the approach proposed in this thesis work have to be improved and deepened. Potential future works are suggested:

- Extend the building of the safety case to more section of ISO 26262. Considering more section of the safety standard could potentially leads to the discovering of problems that were hidden in this thesis work due to the limited part of the standard that has been considered.

- Applying the proposed approach to a larger product line with more safety critical functions (e.g. breaking system) in order to evaluate its scalability. In this way it can be also possible to evaluate if specific tools are needed where a manual approach is unfeasible.

- Deepen the representation of the variation points within the work products of the product line. Finding a feasible and scalable way of representing and managing variation points in different work products, also with the development of a tool chain, could lead to the automatic configuration of the work products including the safety case. Furthermore, it should be considered the possibility to introduce the representation of variation points and the relative variants within the tools under development (e.g. tool for managing requirements), in order to reduce the effort needed to manage numerous versions of the same system.

References

[1] LinkedIn Group, "ISO 26262 Functional Safety," [Online]. Available: http://www.linkedin.com/groups/ISO-26262-Functional-Safety-2308567. [Accessed 3 Jun. 2013].

[2] R. Dardar, "Building a safety case in compliance with ISO 26262," *Master thesis, Mälardalen University, School of Innovation, Design and Engineering,* Västerås, Sweden, 2013.

[3] R. Dardar, B. Gallina, A. Johnsen, K. Lundqvist and M. Nyberg, "Industrial Experiences of Building a Safety Case in Compliance with ISO 26262," *in Proceedings of the Second Workshop on Software Certification (WoSoCER), joint event of the 23rd International Symposium on Software Reliability (ISSRE),* Dallas, Texas, USA, 2012.

[4] B. Gallina, A. Gallucci, K. Lundqvist and M. Nyberg, "VROOM & cC: a Method to Build Safety Cases for ISO 26262-compliant Product Lines," *in Proceedings of the Next Generation of System Assurance Approaches for Safety-Critical Systems (SASSUR), joint event of the 32nd International Conference on Computer Safety, Reliability and Security (SAFECOMP),* Toulouse, France, 2013.

[5] J. Knight, "Safety critical systems: challenges and directions," *in Proceedings of the 24rd International Conference on Software Engineering (ICSE),* Orlando, Florida, USA, 2002.

[6] W. S. Greenwell, J. C. Knight, C. M. Holloway and J. J. Pease, "A Taxonomy of Fallacies in System Safety Arguments," *in Proceedings of the 24th International System Safety Conference (ISSC),* San Diego, California, USA, 2006.

[7] N. Leveson, "Medical Devices: The Therac-25," in *Safeware: System Safety and Computers*, Addison-Wesley, 1995.

[8] European Space Agency, "Ariane 501 Inquiry Board Report," 1996. [Online]. Available: http://esamultimedia.esa.int/docs/esa-x-1819eng.pdf. [Accessed 3 Jun 2013].

[9] Jet Propulsion Laboratory, "Report on the Loss of the Mars Polar Lander and Deep Space 2 Missions," Report, California Institute of Technology, California, USA, 2000.

[10] DVA321, "Safety Critical Systems Engineering," Course at Mälardalen University, School of Innovation, Design and Engineering, Västerås, Sweden. [Online]. Available: http://www.mdh.se/studieinformation/VisaKursplan?kurskod=DVA321. [Accessed 3 Jun. 2013].

[11] "ISO 26262 Road vehicles – Functional safety," *International Standard,* Nov. 2011.

[12] T. Kelly, "A Systematic Approach to Safety Case Management," *in Proceedings of the SAE 2004 World Congress,* Detroit, Michigan, USA, 2004.

[13] F. Redmill and R. Consulting, "Grasping at the Shadow of Safety and Missing the Substance," *in Proceedings of the 6th International Symposium on Programmable Electronic Systems in Safety Related Applications,* Cologne, Germany, 2004.

[14] N. Leveson, "White Paper on the Use of Safety Cases in Certification and Regulation," *Journal of System Safety,* 2011.

[15] I. Habli and T. Kelly, "Process and Product Certification Arguments - Getting the Balance Right," *in Proceedings of 12th IEEE Real-Time and Embedded Technology and Applications Symposium,* New York, New York, USA, 2006.

[16] I. Bate and T. Kelly, "Architectural Consideration in the Certification of Modular Systems," *in Proceedings of the 21 st International Conference on Computer Safety, Reliability and Security (SAFECOMP),* Catania, Italy, 2002.

[17] R. Palin, D. Ward, I. Habli and R. Rivett, "ISO 26262 Safety Cases: Compliance and Assurance," *in Proceedings of the 6th IET International System Safety Conference,* Birmingham, England, 2011.

[18] C. M. Holloway, "Safety Case Notations: Alternatives for the Non-Graphically Inclined?," *in Proceedings of the 3rd IET International Conference on System Safety,* Birmingham, England, 2008.

[19] L. Emmet and G. Cleland, "Graphical notations, narratives and persuasion: a Pliant Systems approach to Hypertext Tool Design," *in Proceedings of the 13th ACM Conference on Hypertext and Hypermedia,* College Park, Maryland, USA, 2002.

[20] T. Kelly and R. Weaver, “The Goal Structuring Notation - A Safety Argument Notation,” *in Proceedings of the Dependable Systems and Networks Workshop on Assurance Cases,* 2004.

[21] GSN Community, “GSN Community Standard Version 1,” Nov. 2011. [Online]. Available: http://www.goalstructuringnotation.info/documents/GSN_Standard.pdf. [Accessed 3 6 2013].

[22] T. Kelly, "Concepts and Principles of Compositional Safety Cases," Research Report commissioned by QinetiQ, 2001.

[23] T. Kelly and J. A. McDermid, "Safety Case Construction and Reuse using Patterns," *in Proceedings of 16th International Conference on Computer Safety, Reliability and Security (SAFECOMP),* York, England, 1997.

[24] Kelly, T.; McDermid, J., "Safety Case Patterns - Reusing Successful Arguments," *in Proceedings of the 16th International Conference on Computer Safety, Reliability and Security (SAFECOMP),* York, England, 1997.

[25] GSN Community, “GSN patterns,” [Online]. Available: http://www.goalstructuringnotation.info/archives/category/resources/patterns. [Accessed 3 Jun. 2013].

[26] I. Habli and T. Kelly, "A Safety Case Approach to Assuring Configurable Architectures of Safety-Critical Product Lines," *in Proceedings of the International Symposium on Architecting Critical Systems (ISARCS),* Prague, Czech Republic, 2010.

[27] P. Clements and L. Northrop, Software Product Lines: Practices and Patterns, Reading, Massachusetts, USA: Addison Wesley, 2001.

[28] P. Heymans and J. C. Trigaux, "Software Product Lines: State of the art," Institut d'Informatique FUNDP, Namur, Belgium, 2003.

[29] S. Thiel and A. Hein, "Modelling and using product line variability in automotive systems," *Software, IEEE,* vol. 19, pp. 66-72, 2002.

[30] D. M. Weiss, “Commonality Analysis: A systematic Process for Defining Families,” in *Proceedings of the 2nd International ESPRIT ARES Workshop on Development and Evolution of Software Architectures for Product Families*, London, England, 1998.

[31] M. Riebisch, “Towards a More Precise Definition of Feature Models,” in *Modelling Variability for Object-Oriented Product*, BookOnDemand Publ. Co, 2003, pp. 64-76.

[32] K. Lee, K. C. Kang and J. Lee, “Concepts and guidelines of Feature Modeling for

Product Line Software Engineering," in *Proceedings of the 7th Reuse Conference on Software Reuse: Methods, Techniques, and Tools: (ICSR7)*, London, England, 2002.

[33] K. Lee and K. C. Kang, "Usage context as key driver for feature selection," in *Proceedings of the 14th international conference on Software product lines: going beyond (SPLC)*, Berlin, Heidelberg, 2010.

[34] S. Friedenthal, A. Moore and R. Steiner, A practical guide to sysml: The Systems Modeling Language, San Francisco, California, USA: Morgan Kaufmann Publishers Inc., 2012.

[35] M. Hause, "The SysML Modelling Language," in *Proceedings of the 15th European Systems Engineering Conference*, Cheltenham, Glos. England, 2006.

[36] F. Puhlmann, A. Schnieders, J. Weiland and M. Weske, "Variability Mechanisms for Process Models," Report, BMBF-Project, 2005.

[37] "Overview of Variability Concepts in EAST-ADL," [Online]. Available: http://www.atesst.org/home/liblocal/docs/ConceptPresentations/08_EAST-ADL_Variability.pdf. [Accessed 25 Jun. 2013].

[38] H. Blom, H. Lönn, F. Hagl, Y. Papadopoulos, M. O. Reiser, C. J. Sjöstedt, C. D. J. and R. Tavakoli Kolagari, *EAST-ADL An Architecture Description Language for Automotive Software-Intensive Systems,* White paper Version M2.1.10, 2012.

[39] O. Lisagor, "Failure logic modelling: a pragmatic approach," *PhD thesis, University of York, York, England,* 2010.

[40] J. V. Earthy, "Hazard and Operability Study as an Approach to Software Safety Assessment," in *Hazard Analysis IEE Colloquium on*, Institution of Electrical Engineers, 1992, pp. 5/1,5/3.

[41] I. Habli, T. Kelly and R. Paige, "Functional Hazard Assessment in Product-Lines - A Model-Based Approach," in *Proceedings of the 1st International Workshop on Model-Driven Product Line Engineering, in conjunction with European Conference on Model-Driven Architecture (ECMDA)*, Twente, The Netherlands, 2009.

[42] S. Baumgart, "Investigations on Hazard Analysis Techniques for Safety Critical Product Lines," in *IDT Workshop on Interesting Results in Computer Science and Engineering (IRCSE)*, Eskilstuna, Sweden, 2012.

[43] J. Dehlinger and R. Lutz, "Software fault tree analysis for product lines," *in Proceedings of the 8th International Symposium on High Assurance Systems Engineering (HASE),* Tampa, Florida, USA, 2004.

[44] R. T. V. Braga, O. T. Junior, K. R. C. Branco, L. D. O. Neris and J. Lee, "Adapting a Software Product Line Engineering Process for Certifying Safety Critical Embedded Systems," *in Proceedings of the 31st international conference on Computer Safety, Reliability, and Security (SAFECOMP),* Magdeburg, Germany, Sep. 2012.

[45] R. T. V. Braga, K. R. L. J. C. Branco, O. T. Júnior and P. C. Masiero, "The ProLiCES Approach to Develop Product Lines for Safety-Critical Embedded Systems and its Application to the Unmanned Aerial Vehicles Domain," *CLEI Electronic Journal,* vol. 15, 2012.

[46] S. Burton and A. Habermann, "Automotive Systems Engineering und Functional Safety: The Way Forward," in *Embedded Real Time Software and Systems (ERTS)*, Toulouse, France, 2012.

[47] I. Habli, I. Ibarra, R. Rivett and T. Kelly, "Model-Based Assurance for Justifying Automotive Functional Safety," *in Proceedings of the SAE World Congress International,* Detroit, Michigan, USA, 2010.

[48] R. Braga, O. J. Trindade, K. R. L. J. C. Branco and J. Lee, "Incorporating certification in feature modelling of an unmanned aerial vehicle product line," *in Proceedings of the 16th International Software Product Line Conference (SPLC),* Salvador, Brazil, 2012.

[49] ATTEST, "http://www.atesst.org," [Online]. [Accessed 21 Aug. 2013].

[50] “Eclipse,” [Online]. Available: http://www.eclipse.org/. [Accessed 25 Jun. 2013].

[51] "EMF Feature Model," [Online]. Available: http://www.eclipse.org/featuremodel/downloads.php. [Accessed 25 Jun. 2013].

[52] “Feature Diagram Editor,” [Online]. Available: https://sdqweb.ipd.kit.edu/wiki/Feature_Diagram_Editor. [Accessed 25 Jun. 2013].

[53] "Feature IDE," [Online]. Available: http://wwwiti.cs.uni-magdeburg.de/iti_db/research/featureide/. [Accessed 25 Jun. 2013].

[54] S. R. Faulk, “Product-Line Requirements Specification (PRS): an Approach and case Study,” *in Proceedings of the 5th IEEE International Symposium on Requirements Engineering (RE),* Toronto, Canada, 2001.

[55] M. Moon, K. Yeom and H. S. Chae, "An Approach to Developing Domain Requirements as a Core Asset Based on Commonality and Variability Analysis in a Product Line," *Software Engineering, IEEE Transactions on,* vol. 31, pp. 551-569, 2005.

[56] M. Riebisch, K. Böllert, D. Streitferdt and B. Franczyk, “Extending The Uml To Model System Families,” *in Proceedings of the 5th World Conference on Integrated Design and Process Technology,* Dallas, Texas, USA, 2000.

[57] J. Dehlinger and R. Lutz, "PLFaultCAT: A Product-Line Software Fault Tree Analysis Tool," *Automated Software Engineering,* vol. 13, pp. 169-193, 2006.

[58] M. Clauss, "Generic Modeling using UML extensions for variability," *in Proceedings of the OOPSLA Workshop on Domain-specific Visual Languages,* Portland, Oregon, USA, 2011.

[59] C. Maga and N. Jazdi, "Survey, Approach and Examples of Modeling Variants in Industrial Automation," *CEAI,* vol. 13, pp. 54-61, 2011.

[60] P. Cuenot, P. Frey, R. Johansson, H. Lönn, M. O. Reiser, D. Servat, R. Tavakoli Kolagari and D. Chen, "Developing Automotive Products Using the EAST-ADL2, an AUTOSAR Compliant Architecture Description Language," *Ingénieurs de l'Automobile (Automobile Engineers),* 2008.

[61] "SysML plugin for Visio," [Online]. Available: http://softwarestencils.com/sysml/. [Accessed 25 Jun. 2013].

[62] “GSN tools,” [Online]. Available: http://www.goalstructuringnotation.info/archives/41#more-41. [Accessed 25 Jun. 2013].

[63] "Assurance Case Editor," [Online]. Available: http://www.goalstructuringnotation.info/archives/268. [Accessed 25 Jun. 2013].

[64] S. Thiel and A. Hein, “Modeling and Using product Line Variability in Automotive Systems,” *Software, IEEE,* vol. 19, pp. 66-72, 2002.