Systematic Derivation of Functional Safety Requirements for Automotive Systems

Kristian Beckers[1], Isabelle Côté[2], Thomas Frese[3], Denis Hatebur[1,2], and Maritta Heisel[1]*

[1] Universität Duisburg-Essen, Germany, Fakultät für Ingenieurwissenschaften, email: {kristian.beckers,maritta.heisel}@uni-due.de
[2] Institut für technische Systeme GmbH, Germany, email: {i.cote,d.hatebur}@itesys.de
[3] Ford Werke GmbH, email: tfrese@ford.com

Abstract. The released ISO 26262 standard for automotive systems requires breaking down safety goals from the hazard analysis and risk assessment into functional safety requirements in the functional safety concept. It has to be justified that the defined functional safety requirements are suitable to achieve the stated safety goals. In this paper, we present a systematic, structured and model-based method to define functional safety requirements using a given set of safety goals. The rationale for safety goal achievement, the relevant attributes of the functional safety requirements, and their relationships are represented by a UML notation extended with stereotypes. The UML model enables a rigorous validation of several constraints expressed in OCL. We illustrate our method using an example electronic steering column lock system.

1 Introduction

The automotive standard for road vehicles ISO 26262 [1], released in 2011, is seen as an automotive industry standard for developing functional safety systems, because it offers the ability to achieve a consistent functional safety process. Its scope covers electronic and electric (E/E) systems for vehicles with a max gross weight up to 3500 kg. Since ISO 26262 is a risk-based functional safety standard addressing malfunctions, its process involves a hazard analysis to determine the necessary risk reduction to achieve an acceptable level of risk. In [2], we described how to define safety goals with an automotive safety integrity level (ASIL) that describes this necessary risk reduction. According to ISO 26262, the next step is to break down these safety goals into functional safety requirements. It has to be justified that the defined functional safety requirements are suitable to achieve the stated safety goals. Functional safety concepts in practice are currently document-based using text processing and drawing tools such as Microsoft

*Part of this work is funded by the German Research Foundation (DFG) under grant number HE3322/4-2 and the EU project Network of Excellence on Engineering Secure Future Internet Software Services and Systems (NESSoS, ICT-2009.1.4 Trustworthy ICT, Grant No. 256980).

Word and Visio. In this paper, we present a systematic, structured and model-based method to define functional safety requirements using a given set of safety goals. The rationale for safety goal achievement, the relevant attributes of the functional safety requirements, and their relationships are represented by using UML notation [3] extended by stereotypes. The UML models enable a rigorous validation of several constraints expressed in the Object Constraint Language (OCL) [4]. Our method is applied to an electronic steering column lock system, serving as illustrative example.

For the break-down of safety goals into functional safety requirements, the ISO 26262 gives no dedicated guidance. It only defines requirements on the content of the documentation. Performing such a break-down is a challenging task because:

- A sound rationale has to be provided.
- Assumptions have to be handled appropriately.
- For the functional safety requirements, the necessary attributes depending on the requirement type have to be defined.
- The functional safety requirements have to be implementable.
- Review activities have to be performed.

In this paper, we propose a structured method based on UML environment models supported by a tool. We assume that an item definition, hazard analysis, risk assessment and safety goals according to ISO 26262 are given (see e.g. [2]). In this paper, we focus on the next step: the creation of a functional safety concept (FSC) in which we show how the functional safety requirements are systematically derived. In the FSC, additionally, requirements may be decomposed in order to lower the ASIL. Furthermore, the functional safety requirements are allocated to elements of a preliminary architecture. These aspects are appropriately described in the ISO 26262 and need no further explanation and improvement and are, therefore, not part of this paper. The contribution of our paper can be summarized as follows:

Rationales are given that show that all safety goals are fulfilled if the requirements are realized, as required by ISO 26262. This will be achieved by using the goal structuring notation with patterns for several solution strategies.

Assumptions are generated based on different sources. It has to be ensured that these assumptions are valid. This is ensured by generating requirements with corresponding descriptions of validation and verification (V&V) activities for them.

Only relevant attributes are described by the developer. This is achieved by classifying the requirements into different categories and by defining, which attributes are required, which are optional, and which should be left out according to the category.

UML profile for **expressing all elements** of a functional safety concept is created in compliance with ISO 26262 making it possible to apply all already mentioned aspects. The profile also provides the basis for validation checks written in OCL.

OCL validation checks concerning consistency and correctness of the functional safety concept are set up. Thus, we provide a **computer-aided technique** to discover errors in the hazard analysis caused by finding inconsistencies or errors in one or more of the UML models.

Functional safety concept document can be generated by the tool, based on the information contained in our UML models. The resulting documentation can then be used for reviewing purposes.

Our paper is organized as follows. The goal structuring notation is introduced in Sect. 2.1. In Sect. 2.2, we give a brief overview of ISO 26262. Our method is presented in Sect. 3. This section also describes our UML profile, which is used to express the functional safety concept. Based on this profile, we define the validation conditions. The tool support is outlined in Sect. 4. We introduce the illustrative example of an electronic steering column lock system as case study in Sect. 5. Section 6 presents related work, while Sect. 7 concludes the paper and gives directions for future work.

2 Background

This section introduces the notation used to derive and justify functional safety requirements (Sect. 2.1. It also provides a short reference to the standard used in this paper (Sect. 2.2).

2.1 Goal Structuring Notation

The Goal Structuring Notation (GSN)[5] - a graphical argumentation notation - explicitly represents the individual elements of any safety argument (goal, strategy, assumption, justification, context, and requirements) and – perhaps more significantly – the relationships that exist between these elements, i.e., how individual requirements are supported by specific strategies, and the assumed context that is defined for the argument.

In the *Functional Safety Concept*, GSN is used to provide an argument for *Functional Safety Requirements* starting from *Safety Goals*, thus also providing the means to check the consistency between Safety Goals and Functional Safety Requirements. In Fig. 1, the GSN elements and their usage for Functional Safety are depicted [6, 7]. The “claim” of the argumentation is the *(Safety) Goal* (e.g. SG03). The *Strategy* expresses the rationale how the goal is addressed by subgoals or functional safety requirements (e.g. ESCL-F-S-Req 03). *Sub-Goals* represent an intermediate step between safety goals and functional safety requirements. Relationships between these elements are expressed with *supported by*,

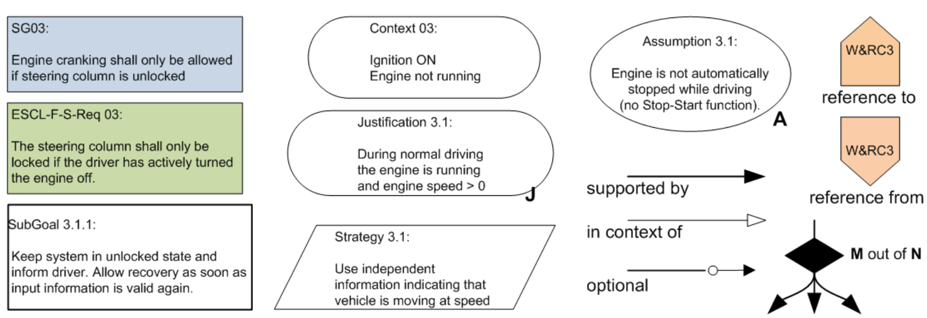

Fig. 1. GSN Notation Overview

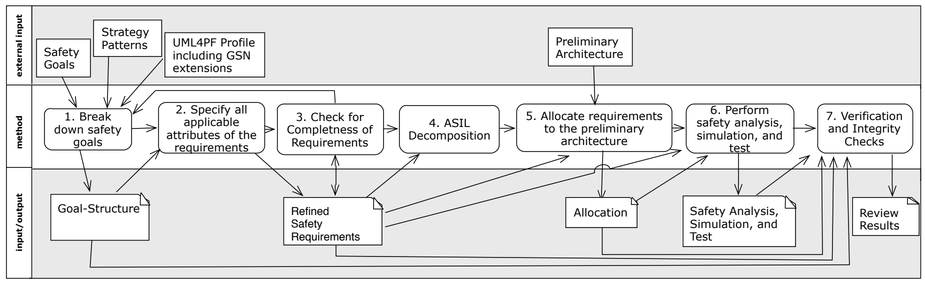

Fig. 2. Method for Functional Safety Concept Creation

optional and alternatives with the element for *M out of N*. For a goal, strategies, subgoals, functional safety requirements, *Context*, *Justifications*, and *Assumptions* can be defined. These relationships are annotated using *in context of*.
Goal structures might reach a size that is hard to fit on a page. To split such a big structure into several smaller ones, we introduced two additional reference elements (see "W&RC3" in Fig. 1).

2.2 ISO 26262

ISO 26262 is a risk-based functional safety standard intended to be applied to safety-related systems that include one or more E/E systems and that are installed in series productions of passenger cars It addresses possible hazards caused by malfunctions of E/E safety-related systems, including the interaction of these systems.

ISO 26262 was derived from the generic functional safety standard ISO/IEC 61508 [8]. It is aligned with the automotive safety life-cycle including specification, design, implementation, integration, verification, validation, configuration, production, operation, service, decommissioning, and management. ISO 26262 provides an automotive-specific risk-based approach for determining risk classes that describe the necessary risk reduction for achieving an acceptable residual risk, called *automotive safety integrity level (ASIL)*.

The possible ASILs are *QM*, *ASIL A*, *ASIL B*, *ASIL C*, and *ASIL D*. The ASIL requiring the highest risk reduction is ASIL D. For functions with ASIL A, ASIL B, or ASIL C, fewer requirements on the development processes, safety mechanisms, and evidences are required. In case of a QM rating, the normal quality measures applied in the automotive industry are sufficient.

3 Method for Functional Safety Concept

We propose a method to create a functional safety concept according to ISO 26262. The aim of the analysis is to break down the generic safety goals into functional safety requirements and allocate them to logical elements of a preliminary architecture. Figure 2 depicts an overview of our method consisting of seven steps. Each step is described in the subsequent paragraphs.

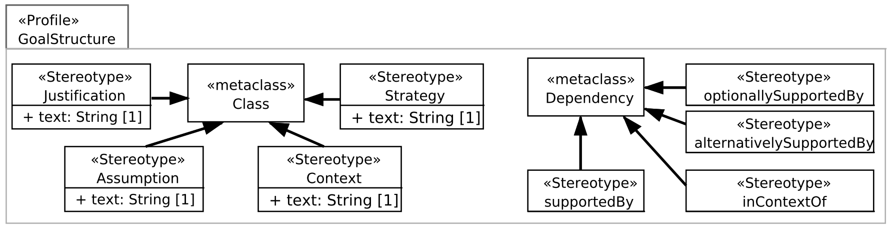

Fig. 3. UML Profile for Goal Structuring Notation Elements

1. Break-down safety goals into functional safety requirements
ISO 26262 requires that the safety goals from the hazard analysis [2] are broken-down into functional safety requirements. This can be documented using the goal structuring notation (see Sect. 2.1). Figure 3 shows the UML profile for the elements of a goal structure. Throughout several projects, it was possible to detect recurring patterns while setting up goal structures. These patterns were transformed into so-called *strategy patterns*. One of these patterns being used in Ford projects, is the *use of independent sources* to obtain certain information, e.g., the vehicle speed and the ignition status can be used for detecting standstill. Further patterns can be found in [9]. The stereotypes for (≪*SafetyGoal*≫, ≪*SubGoal*≫, as well as sub-types of ≪*FunctionalSafetyRequirement*≫) including their respective attributes, are shown in Fig. 4. These elements are explained in more detail in Step 2 of our method. The goal structures document the justification that the functional safety requirements are suitable to address the safety goals obtained from the hazard analysis and risk assessment. They include all assumptions necessary to address the respective safety goal. For better readability, the names of the elements in the goal structure (i.e. safety goal, subgoal, strategy, assumption, context, justification, functional safety requirement or its sub-types) are unique. To verify that, the condition 1M01UE[1] has been formulated (see Tab. 4). According to [6] and [7], not all elements can be connected with each other. The relationships between the different elements are realized as follows:

- classes with the stereotypes ≪*SafetyGoal*≫, ≪*SubGoal*≫, or ≪*Strategy*≫ are connected to classes with stereotypes ≪*SubGoal*≫, ≪*Strategy*≫, or sub-types of ≪*FunctionalSafetyRequirement*≫ by dependencies starting from the former and pointing to the latter. This is checked by condition 1M02DG. Furthermore, we check that two strategies are not directly connected to each other (see Tab. 4, 1M03DS).
- ≪*Justification*≫, ≪*Context*≫, and ≪*Assumption*≫ are connected to ≪*SafetyGoal*≫, ≪*SubGoal*≫, ≪*Strategy*≫ or sub-types of ≪*Functional-SafetyRequirement*≫ by dependencies starting from the former pointing to the latter. This is verified by condition 1M04DC.[2]

[1] The first number refers to the step in the procedure, C is for consistency checks, M is for checks considering correct modeling, G is for generation, the next number is the number of the check within the step, and the last characters are an abbreviation of the description.

[2] In the following, references to validation conditions are given in parentheses.

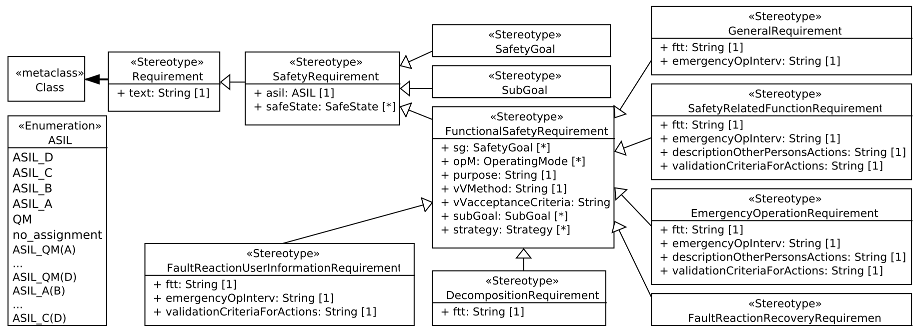

Fig. 4. Elements for Safety Requirements

2. Specify all applicable attributes of the requirements

The requirements developed in Step 1 must to be refined. We support Step 2 with a UML profile that can be used to express the different requirement types. Figure 4 shows the part of our profile that is used to express the different requirement types. A class with the stereotype ≪*Requirement*≫ is used to describe the requirements in general. Safety requirements (≪*SafetyRequirement*≫) are – according to ISO 26262 – special requirements with additional attributes for ASIL and safe states. Safety Goals (≪*SafetyGoal*≫) are top-level safety requirements. A ≪*SubGoal*≫ (not being defined in the ISO 26262) is used in goal structures to structure the argumentation. Functional safety requirements (≪*FunctionalSafetyRequirement*≫) are special safety requirements. They describe the functionalities to achieve the safety goals from a functional perspective without any technical details, such as CAN messages. Each functional safety requirement addresses a set of safety goals (sg), is valid for a given set of operating modes (omM) and should have a purpose (purpose) that may be similar to the strategy or subgoal above. To define functional safety requirements that can be verified, e.g., by testing, the method for verification (vVMethod) and the acceptance criteria (vVAcceptanceCriteria) should be defined. The subgoals or the strategies being supported by the functional safety requirement must be documented. It is important that operating mode, purpose, text, validation and verification method, and acceptance criteria are set for all functional safety requirements (see Tab. 4, 2M01RA). The attributes strategy or subGoal of a ≪*FunctionalSafetyRequirement*≫ can be automatically set based on the information in the goal structure by following the dependencies with the stereotypes ≪*supportedBy*≫, ≪*alternativelySupportedBy*≫ and ≪*optionallySupportedBy*≫ (see Tab. 4, 2M02SG and 2M03SS).

Based on our experience it is helpful to structure the functional safety requirements according to the following categories:

- general requirements,
- safety-related function requirements,
- emergency operation requirements,
- fault reaction: user information requirements,
- fault reaction: recovery requirements, and
- decomposition requirements.

General requirements (≪*GeneralRequirement*≫) could be generic requirements to electronic or electric elements, requirements to elements of other technologies, external measures, or other requirements, e.g., requirements addressing assumptions. For general requirements, it should be possible to define a fault tolerant time (ftt) and the emergency operation interval (emergencyOpInterval). The fault tolerant time defines the period of time between the occurrence of a functional fault and this fault actually becoming dangerous (if it remains undetected). If a safe state cannot be reached by a transition within an acceptable time interval, an emergency operation time interval and a reference to the emergency operation requirement shall be specified. We define general safety requirements for all assumptions to ensure that they are validated or verified. Assumptions are defined

- in the hazard analysis to focus the scope of the analysis to a dedicated vehicle line,
- in the risk assessment on actions of driver or other persons involved to ensure controllability,
- in the rationale for safety goal fulfillment, and
- in the analysis of driver or other persons involved given in the hazard analysis and risk assessment.

Note that it is not necessary to define an ASIL for all general safety requirements, e.g., if they treat external measures or elements of other technologies, no ASIL is required.

We define at least one safety-related function requirement (≪*SafetyRelatedFunctionRequirement*≫) for each safety goal. Safety related-functions include the requirement, the functionality itself, the fault detection requirement and a description of the reaction in case of a detected fault, including transition to a safe state. In addition to the fault tolerant time and the emergency operation interval, a description of actions by the driver or other persons involved (descriptionOfDriverOtherPersonsAction) and validation criteria for these actions (validationCriteriaForActions) can be added. For safety-related function requirements, it is required to specify the ASIL, at least one safe state, and the fault tolerant time (see Tab. 4, 2M04RA).

If an emergency operation interval is specified, we define the corresponding emergency operation requirement (≪*EmergencyOperationRequirement*≫) with the same kind of attributes and conditions as the safety-related function requirement (see Tab. 4, 2M05RA).

If a safe state is entered, usually the driver should be informed. This part of the fault reaction can be defined by user information requirements (≪*FaultReactionUserInformationRequirement*≫). For user information requirements, the fault tolerant time, a description of actions by the driver or other persons involved, and validation criteria for these actions can be added. For user information requirements, it is required to specify at least one safe state, and a description of actions by the driver or other persons involved (see Tab. 4, 2M06RA).

Additionally, the safe state shall be maintained, i.e., the condition for leaving the safe state shall be defined by a fault reaction recovery requirement (≪*FaultReactionRecoveryRequirement*≫). These requirements shall refer to at least one safety goal and the safe state that may be left (see Tab. 4, 2C07RA).

ASIL decomposition requirements (≪*DecompositionRequirement*≫) with fault tolerant time are specified in Step 4. These requirements shall refer to at least one safety goal (see Tab. 4, 2M08RA).

3. Check for completeness of defined requirements

It is important that the functional safety concept is complete. The following criteria can be used to reach this aim:

- for each safe state at least one safety-related function is defined,
- for each assumption at least one general safety requirement is defined,
- for each safe state emergency operation requirement, user information requirements and recovery requirements are defined if, applicable,
- all relevant operating modes are referred to by requirements, and
- requirements necessary to ensure controllability referring to technical means or controls necessary for driver (or other persons involved) actions are identified.

For each safe state, the conditions and the transition to enter this safe state have to be specified. This is achieved by specifying a safety-related function requirement. It can be checked automatically that for each safe state at least one safety-related function requirement is defined (see Tab. 4, 3C01SS) and that for each assumption at least one general safety requirement is defined (see Tab. 4, 3C02AS).

For each safe state and strategy/subgoal-combination, emergency operation requirements, user information requirements, and recovery requirements shall be defined, if applicable. This can be checked by an engineer. The engineer is supported by a table containing all references (see Tab. 4, 3G03SS).

It is important to maintain the consistency of the model of the system to be developed. Therefore, each relevant operating mode shall be referred to by a set of safety-related function requirements. This must be checked by an engineer. The engineer is supported by a table containing all references (see Tab. 4, 3G04OR).

The engineer has to check if all requirements necessary to ensure controllability are identified. These requirements may refer to technical means or controls necessary for the driver (or other persons involved) to perform necessary actions. To perform this step, the engineer has to check the controllability rationales in the risk assessment. The engineer is supported by a table containing controllability rationales (see Tab. 4, 3G05CR). Using this table, the engineer documents appropriate assumptions.

The automated checks mainly cover the consistency of the model. The content of a requirement (e.g., if the requirement text as such is correct and appropriate) has to be verified manually by the engineer.

4. ASIL decomposition

To lower the ASIL for certain components, ASIL decomposition (described appropriately in ISO 26262) can be applied. The necessary requirement category (≪*ASILDecompositionRequirement*≫) has been defined as part of Step 2. In this step, the values for this category are set. The decomposed requirements have a lower ASIL for the technical realization, but the processes have to be established for the original ASIL. This is indicated by providing the original ASIL in parentheses behind the lowered one, e.g. ASIL A(D) (see Fig. 4).

5. Allocation of Requirements

ISO 26262 requires that the functional safety requirements are allocated to the logical blocks in the preliminary architecture. The allocation supports the next document, in which technical safety requirements are generated for dedicated elements (e.g., electronic control units). The allocation can be performed according to safety capabilities, technical complexity of logical blocks, and to commonality of logical blocks with existing requirements. To document this allocation, our UML profile defines the stereotype ≪*LogicalElement*≫ for classes and the stereotype ≪*allocatedTo*≫ for dependencies. The dependencies with the stereotype ≪*allocatedTo*≫ point from classes with the stereotype with a subtype of ≪*FunctionalSafetyRequirement*≫ assigned to classes with stereotype ≪*LogicalElement*≫ (see Tab. 4, 5M01AR).

6. Safety Analysis, Simulation, and Test

ISO 26262 requires to perform safety analysis, simulation, and test. This is beyond the scope of this paper. However, some of the ISO 26262 requirements for this safety analysis are covered by the goal structures set up in Step 1.

7. Verification Review

ISO 26262 requires to perform a verification review of the functional safety concept. This must be performed by a different person who knows the technology of the system-to-be. This is supported by some of the OCL validation constraints in Tab. 4 and the generation of a structured document from the model.

4 Tool Support

```
Dependency.allInstances() ->select(
  getAppliedStereotypes().name ->includes('supportedBy') or
  getAppliedStereotypes().name ->includes('alternativelySupportedBy') or
  getAppliedStereotypes().name ->includes('optionallySupportedBy')
) ->forAll(f|
  (source.getAppliedStereotypes().name ->includes('SafetyGoal') or
   source.getAppliedStereotypes().name ->includes('SubGoal') or
   source.getAppliedStereotypes().name ->includes('Strategy') ) and
  (target.getAppliedStereotypes().name ->includes('SubGoal') or
   target.getAppliedStereotypes().name ->includes('Strategy') or
   target.getAppliedStereotypes().general.name
       ->includes('FunctionalSafetyRequirement')))
```

Listing 1.1. Validation Condition 1M02DG

We used a tool called UML4PF, developed at the University of Duisburg-Essen, and integrated support for the method to create a functional safety concept as described in Sect. 3. After the developer has drawn some diagram(s) using an EMF-based editor, for example Papyrus UML [10] and applied our stereotypes, UML4PF provides him or her with the following functionality: it checks if the developed model is valid and consistent by using our OCL constraints described in Table 4, it returns the location of invalid parts of the model, and it generates documentation that can be used for (manual) validation and review activities.

Basis for the tool is the Eclipse platform [11] together with its plug-ins EMF [12] and OCL [4]. Our UML profile is conceived as an Eclipse plug-in, extending the EMF meta-model. The OCL constraints are integrated directly into the

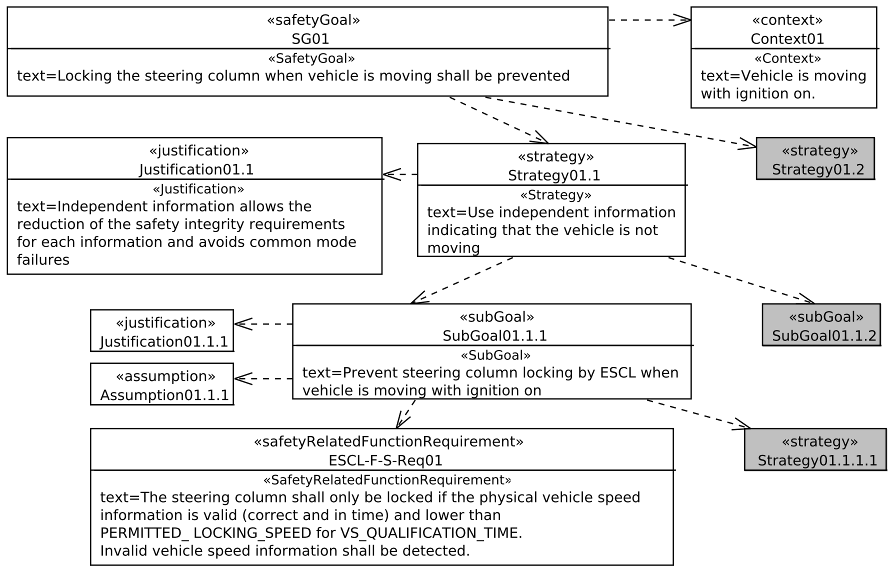

Fig. 5. Goal Structure for SG01 of ESCL

profile. Thus, it is possible to automatically check the constraints using the validation mechanisms provided by Eclipse.

Usually, we consider only one feature at a time in a project. However, it is our believe, that even if all safety related features of a vehicle would be considered in one project, it could be handled by the Eclipse platform running on an appropriate computer.
For example, the OCL expression in Listing 1.1 checks that supporting dependencies connect appropriate elements. To perform the check, it first selects all dependencies (in Line 1) with the either one of the stereotypes ≪*supportedBy*≫, ≪*alternativelySupportedBy*≫ or ≪*optionallySupportedBy*≫ applied (using the EMF keyword getAppliedStereotypes in Lines 2-4). For each of the dependencies matching the stereotypes, it checks if it points from (using the EMF keyword source in Lines 6-8) ≪*SafetyGoal*≫, ≪*SubGoal*≫, or ≪*Strategy*≫ to (using the EMF keyword target in Lines 9-11) ≪*SubGoal*≫, ≪*Strategy*≫, or sub-types of ≪*FunctionalSafetyRequirement*≫ (using the EMF keyword general in Line 11). The other validation conditions given in Table 4 are implemented in a similar way.

5 Case Study

Our case study is an electronic steering column lock (ESCL) system, which was presented at the “VDA Automotive SYS Conference 2012”, June 18/20, 2012, Berlin, Germany and at the VDI Conference “Baden-Baden Spezial 2012”, October 10/11, 2012, Baden-Baden, Germany. Item definition, hazard analysis, risk assessment and the safety goals exist. More details on this topic can be found in [2]. We show the applicability of our method by executing the method steps to the ESCL-example.

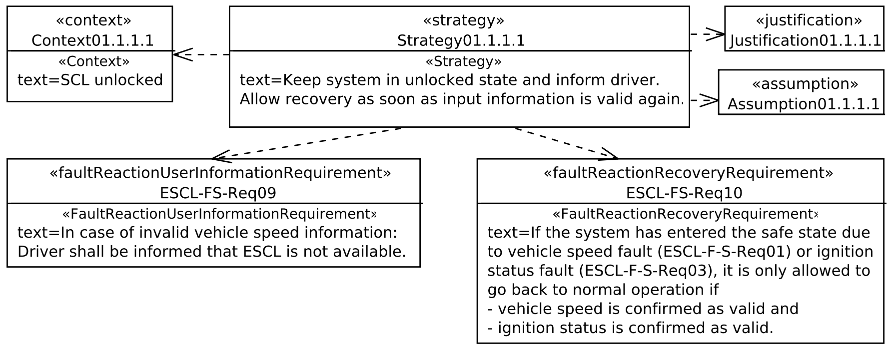

Fig. 6. Goal Structure for Warning and Recovery Concept for SG01 of ESCL

1. Break down safety goals into functional safety requirements
Starting from the safety goals (derivation described in [2]), the goal structures are created. The goal structure in Fig. 5 is created using the pattern "use independent sources" for standstill detection with the appropriate justification (J 01.1) and strategy (S 01.2) to monitor the actuator. Context 01 refers to the situation in which safety is relevant. References to other diagrams are depicted as gray-shaded classes as means to indicate that the diagram is split-up.

Subgoal 01.1.1 and requirement ESCL-F-S-Req01 considering vehicle speed to detect vehicle movement are depicted. Subgoal 01.1.2 considering the ignition status is treated in a different diagram.

Strategy 01.1.1.1 refers to the warning and recovery concept of ESCL-F-S-Req01 and is shown in Fig. 6. In the warning and recovery concept, the context is the safe state that is established by ESCL-F-S-Req01. In the corresponding goal structure given in Fig. 6, a requirement for user information in case of prevented locking and a recovery requirement defining the conditions for entering the normal operation state again, are identified (ESCL-F-S-Req09 and ESCL-F-S-Req10, respectively).

2. Specify all applicable attributes of the requirements
Several requirements have been derived in Step 1. For all of them, it is necessary to specifiy the all relevant attributes. These attributes can be detailed by using a table or to generate such a table. Such a requirement table depicted in Tab. 1. It contains all attributes relevant to ESCL-F-S-Req01.

3. Check for completeness of defined requirements
After defining all attributes of the functional safety concept, it is automatically checked that for each safe state at least one safety-related function is defined and that for each assumption at least one general safety requirement exists by executing Conditions 3C01SS and 3C02AS (see Tab. 4).

To check that for each safe state and strategy/subgoal-combination, all relevant requirement categories have been considered, Tab. 2 was generated automatically (see Tab. 4, 3G03SS) to support the manual completeness check.

An operating mode overview (see Tab. 4, 3G04OR) can be generated from the functional safety concept information. Additionally, a controllability rationale

Safety Req-ID[a]	ESCL-F-S-Req01	**Strategy/Subgoal**	01.1.1
Safety Goal Ref.	SG01, SG02	**Operating Modes**	Steering column unlocked
ASIL Classification (if applicable)	C (D)	**Safe State** (if applicable)	ESCL off; Steering column unlocked
Functional Safety Requirement	The steering column shall only be locked if the physical vehicle speed information is valid (correct and in time) and the absolute value is lower than PERMITTED_LOCKING_SPEED for VS_QUALIFICATION_TIME. Invalid vehicle speed information shall be detected. The PERMITTED_LOCKING_SPEED shall be such that locking below this speed is not dangerous. [b]		
Purpose	To prevent steering column locking while vehicle is moving at speed and steering is required.		
Fault Tolerant Time interval (if applicable)	VS_QUALIFICATION_TIME for vehicle speed faults		
Reduced Functionality interval (if applicable)	n/a		
Functional Redundancies (e.g. fault tolerance) (if applicable)	No		
Description of actions of the driver or other endangered persons (if applicable)	n/a		
Validation Criteria for these actions (if applicable)	n/a		
V&V method	Set vehicle speed > PERMITTED_LOCKING_SPEED while ignition status = ignition off. Set vehicle speed < PERMITTED_LOCKING_SPEED. Fault insertion of vehicle speed signal.		
V&V acceptance criteria	Steering column is not locked until vehicle speed is valid and for VS_QUALIFICATION_TIME below PERMITTED_LOCKING_SPEED.		

Table 1. Attributes of ESCL-F-S-Req01

[a]Req-ID = name of the class

[b]The value for VS_QUALIFICATION_TIME is derived in later phases of the Functional Safety Project (e.g. during creation of the Technical Safety Concept).

overview table (see Tab. 4, 3G05CR) can be generated from risk assessment information. Both tables support reviews by engineers.

4. ASIL decomposition

It is possible to lower the ASIL assigned to SG01. The following decomposition of ASIL D was chosen:

- an ASIL C(D) for no locking in case of vehicle speed,
- an ASIL A(D) for no locking if the ignition status shows that ignition is on, and
- the ASIL decomposition requirement with ASIL D that excludes dependencies between vehicle speed and ignition status.

The decomposition is performed according to ISO 26262.

5. Allocation of Requirements

The functional safety requirements are allocated to the elements of the preliminary architecture. This can be done with UML diagrams as depicted in Fig. 7. From these diagrams, an allocation table as depicted in Table 3 can be generated.

7. Verification Review

To support the reviews, the validation conditions listed in Tab. 4 are executed on the complete case study. These validation conditions check the consistency and correctness of the model. That is, we check

- that all necessary attributes are defined and
- the functional safety concept is complete with respect to the safety goals.

Safety Goal reference	**Safe State**	**Strategy (S) or Sub Goal (SG) reference (optional)**	**Safety Related Functions With this Safe State reference**	**Reduced Functionality reference** (if applicable)	**User Information reference**	**Maintain Safe State / Recovery to Normal Operation reference** (if applicable)
SG01	ESCL off; Steering Column unlocked	1.1.1 (SG)	ESCL-F-S-Req01	n/a	ESCL-F-S-Req09	ESCL-F-S-Req10
		1.1.2 (SG)	ESCL-F-S-Req03			
	Steering Column unlocked and further locking prevented	1.2 (S)	ESCL-F-S-Req05	n/a	ESCL-F-S-Req05b	ESCL-F-S-Req05c
	No engine start allowed due to reduced safety integrity	1.2.1 (SG)	ESCL-F-S-Req05a	n/a	ESCL-F-S-Req05b	ESCL-F-S-Req05c

Table 2. Safe State and Requirement References

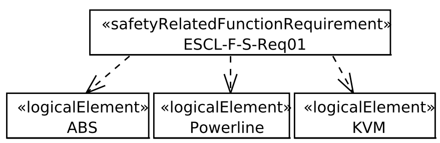

Fig. 7. Allocation of Functional Safety Requirements to Logical Elements

F-S-Req. ID	**Req. text**	**ASIL**	**Subsystem/ Component**							
			KVM	Power Line	K-Line	SCL	ABS	ECM	DIM	Power But.
ESCL-F-S-Req01	...	C (D)	X	X			X			
...										

Table 3. Allocation of Functional Safety Requirements to Logical Elements

6 Related Work

Basir, Denny, and Fischer [9] present goal structures for safety cases in the automotive sector. They do not focus on the technical realization but consider the entire safety process with their documents as entities.

Dittel and Aryus [13] present an overview of V&V activities at Ford Motor Company applied for the lane keeping aid system. This paper also presents elements of the process for functional safety according to ISO 26262, i.e. the analysis activities.

Sinha [14] illustrates an example of a brake-by-wire system for road vehicles including a safety and reliability analysis compliant to ISO 262626. The conclusions derive suggestions for future projects, such as that the system architecture of road vehicles shall support the detection of failures and have the means to still provide desired services until the failures are repaired.

Palin et al. [15] provide guidelines for safety practitioners and researchers to create safety cases compliant to the ISO 26262 standard. The authors propose extensions of the Goal Structuring Notation, patterns, and a number of re-usable safety arguments for creating safety cases. For confidentiality reasons, the authors cannot show example instantiations of their patterns or generic arguments.

Conrad et al. [16] compares software tools that support ISO 26262 certification. The authors identified a list a qualification requirements for selecting ISO 26262 support tools. The publication also contains a report about Conrad et al.'s experience with these tools.

Hillebrand et al. [17] discuss how to develop electric and electronic architectures (EEA) compliant with the ISO 26262 standard. The authors focus on safety requirements during early development phases. Hillenbrand et al. present a method for eliciting safety requirements, and mapping their safety concerns to functions of design artifacts. Previously, Hillebrand et al. [18] proposed a model-based and tool- supported approach for the failure mode and effect analysis (FMEA) of EAAs complaint to ISO 26262. The authors contribute a formalized method for eliciting and analyzing data for a FMEA.

Habli et al. [19] propose a process for model-based assurance for justifying automotive functional safety. They use SysML and GSN as graphical notations. Their goal and ours is similar. We both want to support a method based on ISO 26262 to derive functional safety requirements. In contrast to their work, we use UML, which gives us a broader spectrum of modeling possibilities. Furthermore, we provide tool support for our method and equipped our approach with formal consistency checks on the model. These checks can be automatically checked by our tool. In addition, our way of modeling allows us to trace elements within our models.

Born et al. [20] report on lessons learned from applying a model-based approach for ISO 26262 certification. The authors also discuss the advantages of models instead of text in the ISO 26262 certification process.

7 Conclusions and Future Work

The method presented in this paper has been and currently is applied in several Ford of Europe projects. However, the formal validation conditions and tool support were and are not part of these projects. Both have been developed as contribution of this paper. Still, we are confident that the validation conditions in combination with the tool support ensure at least the same consistency and correctness as the currently used approach, with the benefit of less effort needed. Furthermore, the method is the logical next step to the work presented in [2].

Our contribution has the following main benefits:

- a structured and model-based approach for deriving functional safety concepts for the automotive domain compliant to ISO 26262
- a UML profile to express all required elements for a functional safety concept compliant to ISO 26262
- computer-aided validation of created UML models via executable OCL expressions, e.g., checks for correctness and completeness of the model
- enforcing considering adequate assumptions and safety reasoning by explicitly checking that these are present (by computers) and their soundness (by human experts)

Functional safety concepts in practice are currently document-based using text processing and drawing tools. If the documents are created using a UML tool, the information can be checked for consistency and the document can be created. With our method, a seamless integration into a model-based development process is possible. In the future, we intend to apply our method and tool in different projects. In addition, we plan to focus on technical safety requirements generation and metric derivation.

Step	ID	Condition
1	1M01UE	Names of the elements in the goal structure (safety goal, subgoal, strategy, assumption, context, justification, functional safety requirement or subtype) are unique.
1	1M02DG	Dependencies with the stereotypes ≪*supportedBy*≫, ≪*alternativelySupportedBy*≫ and ≪*optionallySupportedBy*≫ point from classes with the stereotypes ≪*SafetyGoal*≫, ≪*SubGoal*≫, or ≪*Strategy*≫ to ≪*SubGoal*≫, ≪*Strategy*≫, or subtypes of ≪*FunctionalSafetyRequirement*≫.
1	1M03DS	A dependency between 2 strategies is not allowed.
1	1M04DC	A dependency with the stereotype ≪*inContextOf*≫ point from classes with the stereotypes ≪*SafetyGoal*≫, ≪*SubGoal*≫, or ≪*Strategy*≫ to ≪*Justification*≫, ≪*Context*≫ or ≪*Assumption*≫.
2	2M01RA	For each functional safety requirement and their subtypes: the operating mode is required to be set, the purpose, the text, the validation and verification method, and its acceptance criteria is required not to be empty.
2	2M02SG	If the dependency with stereotypes ≪*supportedBy*≫, ≪*alternativelySupportedBy*≫ or ≪*optionallySupportedBy*≫ point from a class with stereotype ≪*SubGoal*≫ to a class with a stereotype being subtypes of ≪*FunctionalSafetyRequirement*≫, its attribute subGoal points to the source of the dependency.
2	2M03SS	If the dependency with stereotypes ≪*supportedBy*≫, ≪*alternativelySupportedBy*≫ or ≪*optionallySupportedBy*≫ point from a class with stereotype ≪*Strategy*≫ to a class with a stereotype being subtypes of ≪*FunctionalSafetyRequirement*≫, its attribute strategy points to the source of the dependency.
2	2M04RA	For a class with the stereotype ≪*SafetyRelatedFunctionRequirement*≫, ASIL, at least one safe state, and fault tolerant are specified.
2	2M05RA	For a class with the stereotype ≪*EmergencyOperationRequirement*≫, ASIL, at least one safe state is referred to, and fault tolerant time is specified.
2	2M06RA	For a class with the stereotype ≪*FaultReactionUserInformationRequirement*≫, at least one safe state is referred to, and a description of actions by the driver or other persons are specified.
2	2C07RA	For a class with the stereotype ≪*FaultReactionRecoveryRequirement*≫, at least one safety goal and one safe state are referred to.
2	2M08RA	For a class with the stereotype ≪*ASILDecompositionRequirement*≫, at least one safety goal is referred to.
3	3C01SS	Each a state or state machine with the stereotype ≪*SafeState*≫ is referred to by a class with the stereotype ≪*SafetyRelatedFunctionRequirement*≫.
3	3C02AS	From each class with the stereotype ≪*Assumption*≫, a dependency with the stereotype ≪*supportedBy*≫ points to a class with the stereotype ≪*GeneralRequirement*≫.
3	3G03SS	For each class with the stereotype ≪*SafetyRelatedFunctionRequirement*≫, all safe states and the related strategies or subgoals are determined. For each combination of safe state and the related strategy or subgoal, references to emergency operation requirements, user information requirements, and recovery requirements are listed in a table.
3	3G04OR	For each state or state machine, the name of the classes with the stereotype ≪*SafetyRelatedFunctionRequirement*≫ are listed in a table. The line is removed, if all substates are referenced or if the containing state is referenced.
3	3G05CR	The controllability rationales from all assessment together with the addressing safety goals are listed in a table.
5	5M01AR	Dependencies with the stereotype ≪*allocatedTo*≫ points from subtype of ≪*FunctionalSafetyRequirement*≫ to ≪*LogicalElement*≫.

Table 4. Validation Conditions

References

1. Int. Organization for Standardization (ISO): Road Vehicles – Functional Safety. ISO 26262 (2011)
2. Beckers, K., Frese, T., Hatebur, D., Heisel, M.: A Structured and Model-Based Hazard Analysis and Risk Assessment Method for Automotive Systems. In: Procs of the 24th IEEE Int. Symposium on Software Reliability Engineering, IEEE Computer Society (2013) 238–247
3. UML Revision Task Force: OMG Unified Modeling Language: Superstructure. Object Management Group (OMG). (May 2010)
4. UML Revision Task Force: OMG Object Constraint Language: Reference (February 2010)
5. Kelly, T.P.: A Systematic Approach to Safety Case Management. In: Procs 28th Symp. on Applied Computing, Detroit, Society for Automotive Engineers (2004)
6. Spriggs, J.: GSN - The Goal Structuring Notation: A Structured Approach to Presenting Arguments. Springer; 2012 edition (2012)
7. Group, G.S.N.W.: GSN community standard version 1 (2011)
8. International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC): Functional safety of electrical/electronic/programmable electronic safety-relevant systems. ISO/IEC 61508 (2000)
9. Basir, N., Denney, E., Fischer, B.: Deriving safety cases for hierarchical structure in model-based development. In: SAFECOMP 2010. LNCS 6351, Springer (2010) 68–81
10. Atos Origin: Papyrus UML Modelling Tool. (Feb 2011) http://www.papyrusuml.org/.
11. Eclipse Foundation: Eclipse - An Open Development Platform. (2011) http://www.eclipse.org/.
12. Eclipse Foundation: Eclipse Modeling Framework Project (EMF) (June 2012) http://www.eclipse.org/modeling/emf/.
13. Dittel, T., Aryus, H.J.: How to 'survive' a safety case according to ISO 26262. In: SAFECOMP 2010. LNCS 6351, Springer (2010) 97–111
14. Sinha, P.: Architectural design and reliability analysis of a fail-operational brake-by-wire system from ISO 26262 perspectives. Reliability Engineering & System Safety (2011) 1349 – 1359
15. Palin, R., Ward, D., Habli, I., Rivett, R.: ISO 26262 safety cases: Compliance and assurance. In: System Safety, 2011 6th IET Int. Conf. on. (2011) 1–6
16. Conrad, M., Munier, P., Rauch, F.: Qualifying software tools according to ISO 26262. In: Proc. Dagstuhl-Workshop Modellbasierte Entwicklung eingebetteter Systeme (MBEES10). (2010)
17. Hillebrand, J., Reichenpfader, P., Mandic, I., Siegl, H., Peer, C.: Establishing Confidence in the Usage of Software Tools in Context of ISO 26262. In: Computer Safety, Reliability, and Security. LNCS. Springer (2011) 257–269
18. Hillenbrand, M., Heinz, M., Adler, N., Matheis, J., Müller-Glaser, K.: Failure mode and effect analysis based on electric and electronic architectures of vehicles to support the safety lifecycle ISO/DIS 26262. In: Rapid System Prototyping (RSP), 2010 21st IEEE International Symposium on. (June 2010) 1–7
19. Habli, I., Ibarra, I., Rivett, R., Kelly, T.: Model-Based Assurance for Justifying Automotive Functional Safety. In: SAE Technical Paper 2010-01-0209. (2010)
20. Born, M., Favaro, J., Kath, O.: Application of ISO DIS 26262 in Practice. In: Procs of the 1st Workshop on Critical Automotive Applications: Robustness & Safety. CARS '10, New York, NY, USA, ACM (2010) 3–6