Deriving Safety Requirements according to ISO 26262 for complex systems: A method applied in the automotive industry

Chapter · June 2017

DOI: 10.1007/978-3-658-18613-5_13

Thomas Frese
Ford Motor Company
8 PUBLICATIONS **27** CITATIONS

Denis Hatebur
University of Duisburg-Essen
58 PUBLICATIONS **542** CITATIONS

A Structured and Model-Based Hazard Analysis and Risk Assessment Method for Automotive Systems

Kristian Beckers, Maritta Heisel
paluno - The Ruhr Institute for Software Technology – University Duisburg-Essen, Germany
Email: {first name.lastname}@paluno.uni-due.de

Thomas Frese
Ford Electrical and Electronic Systems Engineering, Ford Werke GmbH, Germany
Email: tfrese@ford.com

Denis Hatebur
ITESYS Institut für technische Systeme GmbH, Germany
Email: d.hatebur@itesys.de

***Keywords*-hazard analysis, ISO 26262, automotive, safety**

***Abstract*—The released ISO 26262 standard requires a hazard analysis and risk assessment for automotive systems to determine the necessary safety measures to be implemented for a certain feature. In this paper, we present a structured and model-based hazard analysis and risk assessment method for automotive systems. The hazard analysis and risk assessment are based on a requirements engineering process using problem frames. Their elements 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 electronic steering column lock system.**

I. Introduction

The automotive standard for road vehicles ISO 26262 is an automotive industry standard for developing functional safety systems, and offers the ability to achieve a consistent functional safety process. The automotive standard for road vehicles has been released in November 2011. 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 starts with a hazard analysis to determine the necessary risk reduction to achieve an acceptable level of risk. The necessary risk reduction is described by an automotive safety integrity level (ASIL).

Performing such a hazard analysis is a challenging task because

- The result of a hazard analysis needs to be safety goals with an appropriate ASIL that can be the starting point for further development.
- It should be possible to review the hazard analysis within a realistic time period.
- It should be comprehensible for different stakeholders, e.g., engineers, project leaders, managers.
- Hazard analyses of different projects should be comparable.
- In a hazard analysis, all relevant faults or situations need to be considered.

In this paper, we propose a structured method based on UML environment models supported by a tool. We start as required by ISO 26262 with the item definition. According to ISO 26262, the item is a set of functions realized by the system to be built. To support the item definition, we use a UML-based context diagram containing the item, its environment and the system border. The requirements describe the functions of the item by referring to elements of the environment.

The main contributions of our structured method are:

Focused Analysis by **eliminating faults or situations** that are not relevant for a particular hazard analysis. The analysis is based on a set of **fault-type guide-words** and a **hierarchically organized set of situations**. We conduct our reasoning in the beginning of the hazard analysis, because it results in a limited set of relevant faults and situations. This helps safety experts using their time to focus only on this set instead of analyzing all possible faults and situations.

UML Profile for **expressing all elements** of a hazard analysis in compliance with the ISO 26262 standard. The profile provides the basis for object constraint language (OCL) [1] validation checks.

OCL Validation Checks concerning consistency and correctness of the hazard analysis model. Thus, we provide a **computer-aided technique** to discover errors in the hazard analysis caused by inconsistencies or errors in one or more UML diagrams.

Mapping between the table-based representation of the hazard analysis and the UML model. The mapping allows the **import** of existing hazard analyses into the UML model, on which we use our validation conditions. We can also **export** the content of the UML model to a table-based representation for printing and reviewing.

Our paper is organized as follows. The background for our requirements analysis is given in Sect. II-A. In Sect. II-B, we give a brief overview of ISO 26262. Our method is presented in Sect. III. This section also describes our UML profile, which is used to express the hazard analysis and risk assessment. Based on this profile, we define the validation conditions. We introduce an example of an electronic steering column lock system as a case study in Sect. V. Tool support is outlined in Sect. IV. Section VI presents related work, while Sect. VII concludes the paper and gives directions for future work.

II. Background

A. Requirements Analysis

We use a requirements engineering method inspired by Jackson [2]. Requirements can only be guaranteed for a certain context. Therefore, it is important to describe the *environment*, because we build a system to improve something in the world. The environment in which the system to be built (called *machine*) will operate is represented by a *context diagram*.

We use the UML [3] notation with stereotypes defined in the UML profile UML4PF [4] to create a context diagram. Stereotypes give a specific meaning to the elements of a UML diagram they are attached to, and they are represented by labels surrounded by double angle brackets.

A class with the stereotype ≪*machine*≫ represents the thing to be developed, i.e., the machine. The other classes with a ≪*domain*≫ stereotype represent elements in the application environment that already exist. Domains are connected by interfaces consisting of shared phenomena. Shared phenomena may be events, operation calls, messages, and the like. They are observable by at least two domains, but controlled by only one domain, as indicated by an exclamation mark. These interfaces are represented as associations with the stereotype ≪*connection*≫, and the name of the associations contain the phenomena and the domains controlling the phenomena. Figure 4 is an example of a context diagram.

When we state a requirement, we want to change something in the world with the machine to be developed. Therefore, each requirement constrains at least one domain. This is expressed by a dependency from the requirement to a domain with the stereotype ≪*constrains*≫. Such a constrained domain is the core of any problem description, because it has to be controlled according to the requirements. Hence, a constrained domain triggers the need for developing a new system (the machine), which provides the desired control. A requirement may refer to several domains in the environment of the machine. This is expressed by a dependency from the requirement to a domain with the stereotype ≪*refersTo*≫. The referred domains are also given in the requirements description. See Table II for examples.

B. 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 with a max gross weight up to 3500 kg. 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 [5] and was published on 2011-11-11. It is aligned with the automotive safety lifecycle 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 called ASIL D. For functions with ASIL A, ASIL B, or ASIL C, fewer requirements on the development processes, safety mechanisms, and evidences are given in ISO 26262. In case of a QM rating, the normal quality measures applied in the automotive industry are sufficient.

III. A Hazard Analysis and Risk Assessment Method

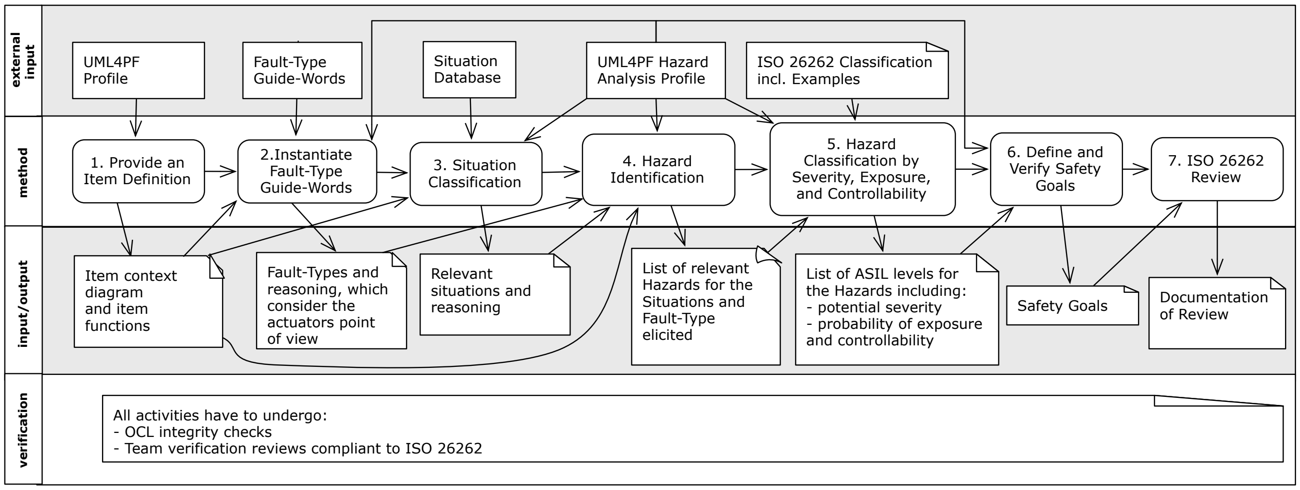
We propose a method for creating a hazard analysis and risk assessment according to ISO 26262. The aim of the analysis is to identify and classify the potential hazards of the item and to formulate safety goals related to the prevention or mitigation of these hazards in order to achieve an acceptable residual risk. The method consists of the following steps, depicted in Fig. 1.

1. Provide an Item Definition: ISO 26262 demands a definition of the item, its basic functionality, and its environment. We use the UML4PF profile to represent this description. The initial description of the item is provided in a context diagram that presents the item and the domains surrounding it, e.g., the driver, see Fig. 4.

The functions of the item are defined by requirements referring to or constraining domains in its environment (see Sect. II-A). Instead of using the stereotype ≪*machine*≫, for the ISO 26262 item definition we use the stereotype ≪*item*≫ to define the domain to be developed.

It is also important to ensure that this step is performed on the right level of detail. It should be avoided to have a too detailed level with too many functions / sub-functions in order to make the hazard analysis assessable.

2. Instantiate Fault-Type Guide-Words: We propose a set of so-called *fault-type guide-words* inspired by the HAZOP standard [6]. The guide-words help the developer to consider all relevant faults. Typical guide words are *no, unintended, early, late, more, less, inverted and intermittent*. Each guide-word has to be instantiated for the functions specified in the *item definition* in the previous step. In the context of a certain function, not all fault-types have to be considered. For example the fault-type “more” is not relevant for a function with a boolean output. For many time-critical functions, a fault-type “late” leads to the same hazard as the fault-type “no”. Usually, it is helpful to start the fault-type consideration from the actuator’s point of view and not from the sensor’s, because the task of the fault-type consideration is not a verification of an existing design. This will be done with appropriate safety analyses, e.g., FMEA [7] and FTA [8, Sect. 7.9], which take place after the ISO 26262 hazard analysis. For all combinations of function



external input
method
input/output
verification
UML4PF Profile
Fault-Type Guide-Words
Situation Database
UML4PF Hazard Analysis Profile
ISO 26262 Classification incl. Examples
1. Provide an Item Definition
2.Instantiate Fault-Type Guide-Words
3. Situation Classification
4. Hazard Identification
5. Hazard Classification by Severity, Exposure, and Controllability
6. Define and Verify Safety Goals
7. ISO 26262 Review
Item context diagram and item functions
Fault-Types and reasoning, which consider the actuators point of view
Relevant situations and reasoning
List of relevant Hazards for the Situations and Fault-Type elicited
List of ASIL levels for the Hazards including:
- potential severity
- probability of exposure and controllability
Safety Goals
Documentation of Review
All activities have to undergo:
- OCL integrity checks
- Team verification reviews compliant to ISO 26262

Figure 1: A Method for Hazard Analysis and Risk Assessment compliant to ISO 26262

and fault, we describe how the system behaves in presence of the malfunction.

We support this step with a UML profile that can be used to express the different artifacts. Figure 2 shows the part of the profile that is used to express the faults of an actuator constrained by a functional requirement. A class with the stereotype ≪*fault*≫ is used to describe the faults. Each fault may be in one of the fault-types *no, unintended, early, late, more, less, inverted and intermittent*. For each requirement, all fault-types are checked if they have to be considered, are not possible, or are covered by other faults (this is checked by condition 2C01CF in Tab. I[1]) A dependency with the stereotype ≪*fails*≫ is used to show the relation between requirements and faults (see Tab. I, 2M02HF). Additionally, fault constraints can be described. For each considered fault, we describe the effect on the system level (see Tab. I, 2C03EF) and not on component or vehicle level. On the system level, the elements of the item are visible, e.g., actuators. In vehicle level descriptions, only phenomena that can be observed or controlled by the driver or other persons are used. The component level contains descriptions of internal interfaces, e.g., CAN [9] messages. For faults rated not to be considered, either a description why it is not relevant or a reference to at least one other fault using the attribute "covered by" has to be specified (see Tab. I, 2C04DC).

3. Situation Classification: From project-experience, examples in the ISO 26262, national and international working groups, a hierarchically organized list of situations is created. Using this list, we rate if a situation is relevant for the described item with its requirements or not. If a more abstract situation is rated, it is not necessary to rate the special situations (see Tab. I 3C01RE). If a situation is rated as not being relevant, either a reference to another situation that includes this situation is given, or a rationale is provided (see Tab. I 3C02RR). The hierarchically organized list of situations is updated if new aspects are identified in projects in order to reduce the risk of forgetting hazardous situations.

In the UML profile, a number of situations are defined as stereotypes. Situations are, e.g., ≪*standstill Situation*≫, ≪*maneuverSituation*≫, and ≪*driving Situation*≫. Special standstill situations are ≪*standstillEngineOffSituation*≫ (e.g., in parking lot) and ≪*standstillEngineOnSituation*≫. Special maneuver situations are, e.g., ≪*parkingManeuverSituation*≫ and ≪*drivingBackwardSituation*≫. Special driving situations are on the one hand driving activities like ≪*brakingSituation*≫, ≪*accelerationSituation*≫, ≪*steeringSituation*≫, and ≪*rollingSituation*≫, and on the other hand driving areas like ≪*citySituation*≫, ≪*countryRoadSituation*≫, and ≪*highwaySituation*≫.

4. Hazard Identification: For each fault/function combination, all situations that could lead to a potential hazard are identified in the list of situations being relevant. We describe the effect on the vehicle level, i.e., what behavior could occur in case of a potential item's malfunction. Based on the effect on the vehicle level, we describe the hazards and possible consequences. Hazards are defined in terms of the conditions or events that can be observed at the vehicle level (e.g, by the driver). A verbal description of consequences without ranking is given. In this step, also assumptions (e.g., on driver actions to maintain controllability) should be considered. Due to the complexity of documenting assumptions, we do not discuss assumptions in this paper.

A hazard is ≪*causedBy*≫ a set of faults and refers to situations ≪*when*≫ it can occur (see Table I, 4M01CB,

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

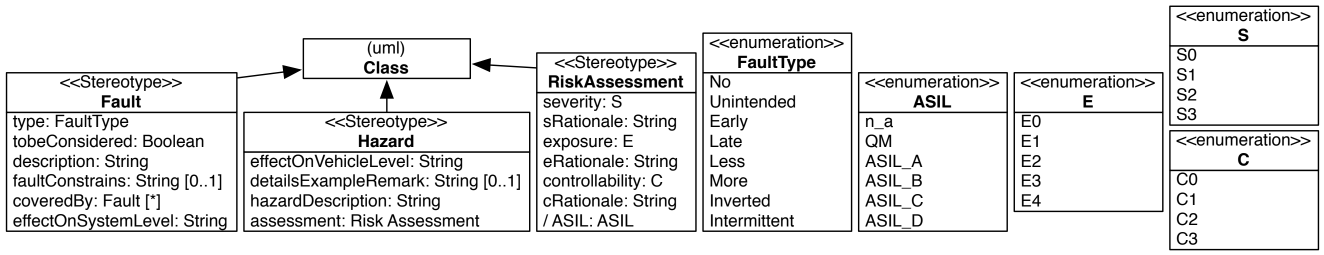


Figure 2: Fault, Hazard, Assessment

Table I: Validation Conditions

Step	ID	Condition
2	2C01CF	All fault-types are considered for functional requirements in the item definition that constrain a domain.
2	2M02HF	Dependencies with the stereotype ≪*fails*≫ only point from classes with the stereotype ≪*requirement*≫ to classes with the stereotype ≪*fault*≫.
2	2C03EF	If a fault is rated as to be considered, the effect on system level is described (string not empty).
2	2C04DC	If a fault is rated as not to be considered, a reference to at least one other fault using the attribute covered by is provided or a description with a rationale is provided.
3	3C01RE	All situation types are rated if they are relevant for the feature or not. If a more general situation is rated, it is not necessary to rate the detailed (derived) situations.
3	3C02RR	If a situation is rated as not being relevant, a rationale is given (rationale string shall not be an empty string) or in the attribute covered by a reference to at least one other situation is provided.
4	4M01CB	A dependency with the stereotype ≪*when*≫ only points from a class with the stereotype ≪*Hazard*≫ to a class with the stereotype ≪*Situation*≫.
4	4M02WH	A dependency with the stereotype ≪*causedBy*≫ only points from a class with the stereotype ≪*Hazard*≫ to a class with the stereotype ≪*Fault*≫.
4	4C03RW	Each situation being relevant is referenced by at least one hazard (using ≪*when*≫-dependency).
4	4C04CC	Each fault of a domain being marked as “to be considered” is referenced by at least one hazard (using ≪*causedBy*≫-dependency).
4	4C05WC	Each Hazard has at least one ≪*when*≫-dependency and at least one ≪*causedBy*≫-dependency.
5	5C01SR	For a severity below S3 a rationale is provided.
5	5C02SA	The same rationale for a severity does not lead to different rating in another assessment.
5	5C03ER	For an exposure below E4 a rationale is provided.
5	5C04EA	The same rationale for an exposure does not lead to different rating in another assessment.
5	5C05CR	For a controllability below C3 a rationale is provided.
5	5C06CA	The same rationale for a controllability does not lead to different rating in another assessment.
6	6M01SH	A dependency with the stereotype ≪*addresses*≫ only points from a class with the stereotype ≪*safetyGoal*≫ to a class with the stereotype ≪*Hazard*≫.
6	6C02QM	All ≪*hazard*≫s rated with QM are addressed by at least one ≪*requirement*≫.
6	6C03HA	All ≪*hazard*≫s rated with ASIL A-D shall be addressed by at least one ≪*safetyGoal*≫.
6	6C04SA	The ASIL of the ≪*safetyGoal*≫ is equal to the highest ASIL of the addressed ≪*hazard*≫s.

4M02WH). This is expressed by stereotyped dependencies. It is important that each relevant situation is referenced by at least one hazard (see Table I, 4C03RW), and each of the faults of a domain that has to be considered is referenced by at least one hazard (see Table I, 4C04CC). Additionally, each Hazard has at least one ≪*when*≫-dependency and at least one ≪*causedBy*≫-dependency (see Table I, 4C05WC).

To describe the hazard, we use the stereotype ≪*hazard*≫ as depicted in Fig. 2 and start with the description of the effect on vehicle level. We may give details, examples, or remarks, and provide a description of the hazard that includes the situations and the fault effect. The hazard refers to the risk assessment to be performed as described in the next paragraph.

5. Hazard Classification by Severity, Exposure, and Controllability: The objective of the hazard classification is to assess the level of risk reduction required for the hazards. To classify the hazard, the following steps need to be performed according ISO 26262:

1) Estimate the potential severity and provide a rationale. ISO 26262 classifies the potential severity with the classes S0 (no injuries), S1 (light and moderate injuries), S2 (severe and life-threatening injuries, survival probable), and S3 (life-threatening injuries, fatal injuries).
2) Estimate the probability of exposure and provide a rationale. ISO 26262 classifies the exposure with the classes E0 (incredible, e.g., earthquake), E1 (very low probability, e.g., vehicle being towed), E2 (low probability, e.g., snow and ice on road), E3 (medium probability, e.g., heavy traffic with stop and go), E4 (high probability, e.g., highway).
3) Estimate the controllability and provide a rationale. ISO 26262 classifies the controllability with the classes C0 (controllable in general, e.g., maintain intended driving path in case of unexpected radio volume increase), C1 (simply controllable, e.g., brake to slow down/stop the vehicle in case of blocked steering column when starting the vehicle), C2 (normally controllable, e.g., maintain intended driving path in case of failure of ABS during emergency braking, C3 (difficult to control or uncontrollable, e.g., stay in lane in case of failure of ABS when braking on low friction road surface while executing a turn).

The description of and examples for the classes are taken from the standard [10, Part 3, Appendix B]. The risk assessment is documented in a class with the stereotype ≪*RiskAssessment*≫ as depicted in Fig. 2. This stereotype has the attributes S, E, and C, each of them typed with enumerations representing the classes. Additionally, it has the attributes sRationale, eRationale, and cRationale of type String that are used to provide a rationale for the selected class. For a severity below S3, an exposure below E4, and a controllability below C3, a rationale has to be provided (see Table I, 5C01SR, 5C03ER, 5C05CR). The same rationale cannot lead to different rating in other assessments (see Table I, 5C02SA, 5C04EA, 5C06CA). For example, the controllability rationale 'no lateral control by steering is possible' cannot lead to C3 in one assessment and to C2 in another assessment.

Based on these estimations, the ASIL is determined automatically according to the corresponding ISO 26262 table. For example, a rating of S3, E4, and C3 leads to ASIL D. If one of the parameters is reduced to the lower class, ASIL C is derived. If the parameters are reduced more, ASIL B, ASIL A or QM is derived. In case of S0, E0, or C0, no ASIL is assigned and n_a is inserted into the rating attribute ASIL.

6. Define and Verify Safety Goals: Safety requirements are special requirements with the attributes ASIL, safe state, and fault tolerance time (see Fig. 3). The ASIL is a measure of necessary risk reduction. The safe state is a state that shall be entered to avoid a hazard. The fault tolerance time is the time an actuator state can be unsafe before the situation becomes hazardous, e.g., an undue brake intervention may have a fault tolerance time of 100 ms in certain situations. Safety requirements are defined on different levels. A safety goal is a top-level safety requirement based on the hazards identified in this analysis. Functional safety requirements are derived from the safety goals. Technical safety requirements are derived from the functional safety requirements and specified for all components of the item considering the concrete architecture.

Safety goals have to be clear and precise, do not contain technical details, but have to be implementable by technical means (e.g. avoid referring to non-measurable data). Like functional requirements, safety goals refer to domains and constrain at least one domain.

ISO 26262 requires that at least one safety goal is assigned to each hazard rated as ASIL A, B, C or D (see Tab. I 6C03HA). It is not necessary to define safety goals for hazards rated as "QM" or "no assignment", but hazards rated with QM shall be addressed by at least one requirement (see Tab. I 6C02QM).

One safety goal can address several hazards (see Tab. I 6M01SH). A hazard can be addressed by more than one safety goal. ISO 26262 requires that if a safety goal can be achieved by transitioning to or by maintaining one or more safe states, then the corresponding safe states are specified.

In our tool, the ASIL of the safety goal is set automatically to the highest ASIL of the addressed hazards, as required by ISO 26262. This is also verified by a validation condition (see Table I, 6C04SA). For each safety requirement, and therefore also for the safety goals, additionally the fault tolerance time has to be specified.

7. ISO 26262 Review: ISO 26262 requires that the results of the hazard analysis and risk assessment shall be reviewed by an independent person from a different department or organization, i.e., independent from the department responsible for the hazard analysis regarding management, resources and release authority.

During the entire creation phase of the hazard analysis and risk assessment, correctness and consistency of intermediate results are checked by

- OCL integrity checks as described in Table I.
- Team verification reviews compliant to ISO 26262.

It is possible, to directly generate a document from the model. The safety goals and faults referenced by or referencing multiple other elements will appear in more then one table row. For the opposite direction, a tool has to detect, which entries occur in more then one row.

IV. Tool Support

We have developed a tool called UML4PF to support the requirements engineering process sketched in Sect. II, as well as the hazard analysis and risk assessment process described in Sect. III.

After the developer has drawn some diagram(s) using some EMF-based editor, for example Papyrus UML [11], UML4PF provides him or her with the following functionality: it checks if the developed model is valid and consistent by using our OCL constraints as described in Table I, and it returns the location of invalid parts of the model.

Basis for our tool is the Eclipse platform [12] together with its plug-ins EMF [13] and OCL [1]. Our UML-profile is conceived as an Eclipse plug-in, extending the EMF metamodel. We store the data in the profile in XMI-format. We use UML with our own profile that extends UML with the ability to express requirements in a similar way as the SysML profile [14] extends UML. If a SysML model with blocks describing the context and requirements was given, we could use the same approach by extending SysML by the missing stereotypes and constraints.

We store all our OCL constraints in one file in XML-format. They are directly checked using the OCL executor, which is part of EMF.

For example, the OCL expression in Listing 1 checks the condition that if a fault is rated as not to be considered, a reference to at least one other fault using the attribute covered by is provided or a description with a rationale is provided. To perform the check, it first selects all classes with the stereotype ≪*Fault*≫ applied (using the EMF keyword getAppliedStereotypes in Line 1), and for all these faults, it

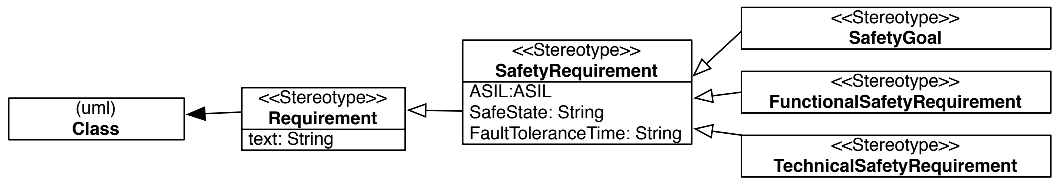


Figure 3: Requirements

- sets st to be the stereotype of the class (line 2).
- Using st, it retrieves the value of the boolean attribute toBeConsidered (line 4).
- If the attribute is not *true* (line 4),
- it is checked that a description (that should contain the rationale) is not undefined (line 5) and not empty (line 6), or
- a reference to at least one other fault is given (line 7).

```
1  Class.allInstances()
       ->select(getAppliedStereotypes().name
       ->includes('Fault')) ->forAll(f|
2    let st: Stereotype = f.getAppliedStereotypes()
         ->select(name = 'Fault') ->asSequence()
         ->first()
3    in
4      not f.getValue(st,'toBeConsidered')
           .oclAsType(Boolean) implies
5      (( not f.getValue(st,'description')
           .oclAsType(String) .oclIsUndefined() and
6      f.getValue(st,'description')
           .oclAsType(String) <> '') or
7      f.getValue(st,'coveredBy') ->size()>0)
8  )
```

Listing 1: Validation Condition 2C04DC

```
1  Class.allInstances()
       ->select(getAppliedStereotypes().name ->
       includes('Requirement'))->forAll(req|
2    let faults: Set(Class) = req.oclAsType(Class)
         .clientDependency
         ->select(getAppliedStereotypes() .name
         ->includes('fails')) .target
         .oclAsType(Class)->asSet()
3    in
4    faults ->size()>0 and
5    (let st: Stereotype =
         faults.getAppliedStereotypes()
         ->select(name = 'Fault') ->asSequence()
         ->first()
6    in
7    let ty: Set(EnumerationLiteral) = faults
         .getValue(st,'type')
         .oclAsType(EnumerationLiteral) ->asSet()
8    in of
9      ty.name -> includes('No') and
10     ty.name -> includes('Unintended') and
11     ty.name -> includes('Early') and
12     ty.name -> includes('Late') and
13     ty.name -> includes('Less') and
14     ty.name -> includes('More') and
15     ty.name -> includes('Inverted') and
16     ty.name -> includes('Intermittent'))
17 )
```

Listing 2: Validation Condition 2C01CF

The OCL expression in Listing 2 checks if all fault-types are considered for each functional requirement, i.e., an object containing a rationale or reference exists for each fault-type. For all requirements (line 1), the set of fault-types the requirement refers to with a *≪fails≫* dependency are determined and named faults (line 2 and 3). It is checked that more than 0 faults are in this set (line 4), and by using the fault stereotype st (lines 5 and 6) it is checked that all types of faults (no, unintended, ...) are in the set faults (lines 7-17). The type of the fault is an enumeration attribute of the stereotype *≪Fault≫* named type. The set of fault-type attributes ty is retrieved using the EMF keyword getValue in line 7.

The other validation conditions given in Table I are implemented in a similar way.

V. Case Study

We illustrate our method on an example of an electronic steering column lock (ESCL) system, which was presented at the "VDA Automotive SYS Conference 2012", June 18/20, 2012, in Berlin, Germany.

1. Provide an Item Definition: The main function of the ESCL is to provide lock and unlock commands to the lock actuator automatically to enhance theft protection for vehicles with a power button instead of a standard key. The context diagram in Fig. 4 shows the item, i.e., the ESCL and the elements in the environment, namely the the driver, the lock actuator, and the vehicle. The item controls the lock and the unlock commands, and the lock actuator observes these phenomena. The driver presses the power button to crank or stop the engine. The vehicle moves at a certain speed and therefore controls the phenomenon Speed.

Table II shows the functional requirements of the ESCL. For both requirements R01 and R02, it shows which domains of the item's environment are constrained (see column *≪constrains≫*) and which domain the requirements are referred to (see column *≪refersTo≫*): The lock actuator is constrained. The driver and the vehicle are referred to since they together provide necessary information to deduce if the driver wants to drive or not.

2. Instantiate Fault-Type Guide-Words: To start with the hazard analysis, we look at the lock actuators constrained by the requirements $R01$ and $R02$, and investigate both functional requirements according to the guide-words.

Figure 5 shows some of the fault types assessed for $R01$. For the guide-word 'no', the fault is that the ESCL does not lock in situations where it is expected. For the guide-word 'unintended', the fault is that the ESCL locks in situations where it is not allowed. For the guide-word 'early', the fault

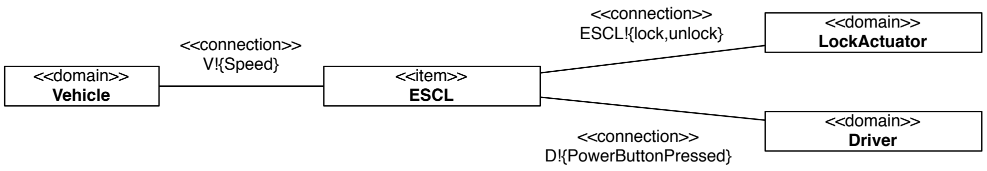


Figure 4: ESCL Context Diagram

Table II: ESCL Requirements

No	**Text**	≪***constrains***≫	≪***refersTo***≫
R01	The steering column shall be locked, when the driver wants to immobilize the vehicle.	LockActuator	Driver, Vehicle
R02	The steering column shall be unlocked, when the driver wants to drive.	LockActuator	Driver, Vehicle

is the same as described in fault unintended_lock. For the guide-word 'late', the fault is either no problem, or in case of a long delay the same as described in fault no_lock.The faults related to all other guide-words are either mapped to 'no', 'unintended', or are not relevant since the locking is a binary decision and cannot be 'less' or 'more'.

Additionally, we have to describe the effect on system level. For example, for unintended locking, the effect on the system level is that the ESCL locks the steering column. The effect on the vehicle level is that the vehicle is not steerable. For $R02$, the procedure is the same.

3. Situation Classification*:* The situations are classified according to the item's functionality. In Fig. 6, the situation classification using our profile is depicted.

The situation 'driving at speed' is classified as being relevant, because a hazard may occur if the vehicle is moving and the steering column is locked. The situation 'maneuver' (including, e.g., parking, driving backwards) is marked as being not relevant, because for the ESCL system the maneuvering hazards are the same as for 'driving at speed'. The situation 'standstill' is classified as being relevant, because a hazard may occur if the vehicle is 'parked', and the steering column is not locked. The effort necessary for the situation classification is reduced, because it is not necessary to rate these detailed situations. The situations 'being towed' and 'rolling' are classified as being relevant because they consider the system state where the engine is off.

4. Hazard Identification*:* In the next step, the hazards are identified. For this reason, all considered faults are combined with all situations where the fault leads to a problem. These are the ones having the attribute toBeConsidered=true. It is possible to have more than one fault that causes the hazard, and the hazard can be in place in different situations. Some combinations are not needed in the hazard analysis, e.g., the situation standstill does not need to be combined with the fault of unintended locking, because locking is intended in this situation.

Fig. 7 shows the hazard that can occur when the vehicle is moving at speed. It may be caused by unintended steering column locking. To describe the hazard, first the effect on vehicle level is described. For the previously described fault, the effect on vehicle level is that the steering is locked and the vehicle is not steerable. The hazard is the "loss of steering control (locked steering) when driving at speed". The hazard refers to the risk assessment performed in the next step.

5. Hazard Classification by Severity, Exposure, and Controllability*:* The above-mentioned hazard is rated according to its severity, exposure, and controllability as depicted in Fig. 7. The highest severity level S3 is chosen, because a locked steering column lock at speed can lead to death or life-threatening injuries when the vehicle hits, e.g., obstacles near the road, pedestrians, oncoming traffic, or obstacles on the track. The exposure level is E4, because steering is necessary in the mentioned situations and in all situations with high speed, which are more than 10 % of the driving time. The highest controllability level C3 is chosen since no lateral control by steering is possible. The driver can only intervene by braking, but in case of high speed the driver cannot avoid the consequences of the hazard. Hence, ASIL D is automatically deduced from this classification.

6. Define and Verify Safety Goals*:* The described hazard can be addressed by a safety goal as depicted in Fig. 8. The same safety goal can also address other hazards (not shown here). To prevent the hazard, the safety goal is that "locking of the steering column when the vehicle is moving shall be prevented". This safety goal refers to the vehicle since it indicates if the vehicle is moving and constrains the steering column lock by preventing the lock. Since we cannot accept steering column locking even for a certain time, the fault tolerance time is not applicable. For other safety goals, e.g., prevention of braking intervention, the fault may be acceptable for a short time period. We model the relations between faults, hazards, and safety goals, depicted in Fig. 9. The information in this model can be converted into a table (see Tab. III) for documentation purposes. The safety goals (e.g., PreventLocking) and faults (e.g., unintended_lock) referenced by or referencing multiple other elements will appear in more than one table row. Since the hazard Theft (see Fig. 9) has no ASIL A, B, C or D assigned, it is not addressed by a safety goal but by the requirement EnsureLocking.

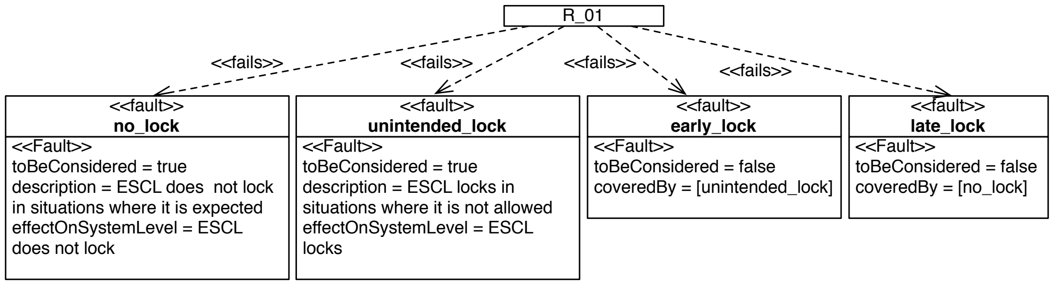

R_01
<<fails>>
<<fails>>
<<fails>>
<<fails>>
<<fault>> no_lock
<<Fault>>
toBeConsidered = true
description = ESCL does not lock in situations where it is expected
effectOnSystemLevel = ESCL does not lock
<<fault>> unintended_lock
<<Fault>>
toBeConsidered = true
description = ESCL locks in situations where it is not allowed
effectOnSystemLevel = ESCL locks
<<fault>> early_lock
<<Fault>>
toBeConsidered = false
coveredBy = [unintended_lock]
<<fault>> late_lock
<<Fault>>
toBeConsidered = false
coveredBy = [no_lock]

Figure 5: ESCL faults

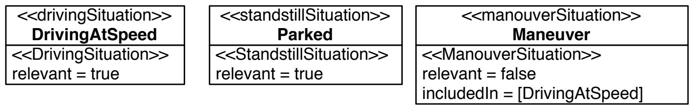

<<drivingSituation>> DrivingAtSpeed
<<DrivingSituation>>
relevant = true
<<standstillSituation>> Parked
<<StandstillSituation>>
relevant = true
<<manouverSituation>> Maneuver
<<ManouverSituation>>
relevant = false
includedIn = [DrivingAtSpeed]

Figure 6: ESCL Situations

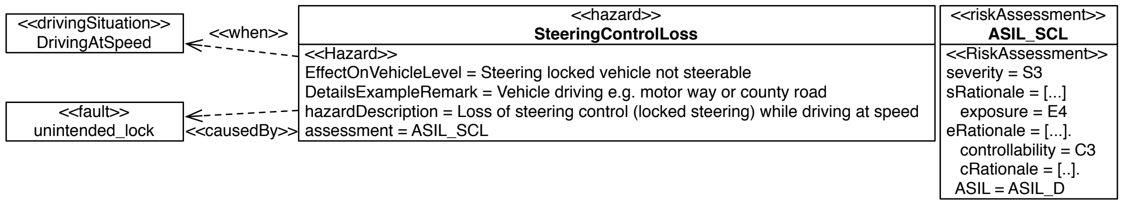

<<drivingSituation>> DrivingAtSpeed
<<when>>
<<fault>> unintended_lock
<<causedBy>>
<<hazard>> SteeringControlLoss
<<Hazard>>
EffectOnVehicleLevel = Steering locked vehicle not steerable
DetailsExampleRemark = Vehicle driving e.g. motor way or county road
hazardDescription = Loss of steering control (locked steering) while driving at speed
assessment = ASIL_SCL
<<riskAssessment>> ASIL_SCL
<<RiskAssessment>>
severity = S3
sRationale = [...]
exposure = E4
eRationale = [...].
controllability = C3
cRationale = [..].
ASIL = ASIL_D

Figure 7: ESCL Hazard Example

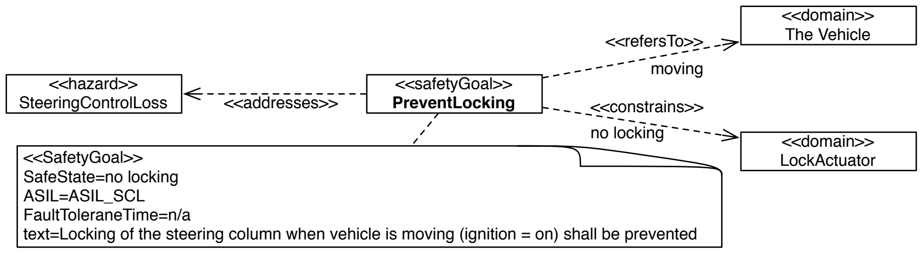


Figure 8: ESCL Safety Goal

7. ISO 26262 Review: To support the reviews, the validation conditions listed in Table I are executed on the complete case study. These validation conditions check the consistency and correctness of the model. That is, we check

- whether each relevant functional requirement in the item definition is considered,
- whether the hazard and risk assessment is aligned with the supplier's assessment, and
- whether the hazard and risk assessment is consistent with ISO 26262 description.

VI. Related Work

We are not aware of any publications about a structured and model-based hazard analysis and risk assessment for automotive systems equipped with integrity checks.

Two hazard analysis methods are compared by Törner et al. [15]. The paper shows that the adapted functional failure analysis (FFA) is less time-consuming than the method of the European Space Agency (ESA method). The method presented is this paper is based on the results of [15].

The entire safety lifecycle including hazard analysis and risk assessment is presented by Baumgart [16]. Our method can complement the hazard analysis of Baumgart's safety lifecycle.

The Safety Management System and Safety Culture Working Group provides guidance on hazardbb identification by different means, e.g., brainstorming, HAZOP, checklists, FMEA [17]. Their results are considered in the method presented in this paper.

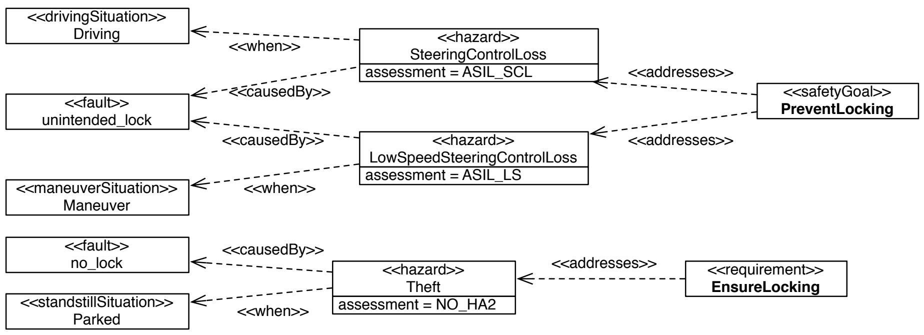


Figure 9: Safety Goal - Hazard - Fault Relataions

Table III: Safety Goal - Hazard - Fault Relataions

Fault	**Situation**	**Hazard**	**Risk Assessment**	**Safety Goal / Requirement**
unintended_lock	Driving	SteeringControlLoss	ASIL_SCL	PreventLocking
unintended_lock	Manouver	LowSpeed SteeringControlLoss	ASIL_LS	PreventLocking
no_lock	Parked	Theft	NO_HA2	EnsureLocking

Jesty et al. [18] give a guideline for the safety analysis of vehicle-based systems, including system analysis, hazard identification, hazard analysis, identification of safety integrity levels, FMEA, and fault tree analysis. Their work also uses the HAZOP guide-words, but they focus on the safety integrity level as defined in the IEC 61508 and not on the ASIL from ISO 26262. Jesty et al. additionally address FMEA and fault tree analysis for analysing existing systems, but do not consider a model or validation conditions.

In contrast to our work, which focuses on the determination of necessary risk reduction, following papers describe model-based approaches specific for later development phases, when the system is already designed and not the determination of necessary risk reduction:

Papadopoulos and Grante [19] propose a process that addresses both cost and safety concerns and maximizes the potential for automation to address the problem of increasing technological complexity. It combines automated safety analysis with optimization techniques.

Li and Zhang [20] present a comprehensive software hazard analysis method, which applies a number of hazard analysis techniques, and the proposed method is applied to a software development process of a control system. The described method for hazard analysis is similar but less detailed than ours.

Mehrpouyan [21] proposes a model-based hazard analysis procedure (based on SysML models) for the early identification of potential safety issues caused by unexpected environmental factors and subsystem interactions within a complex safety-critical system. The proposed methodology additionally maps hazard and vulnerability modes to specific components in the designed system and analyzes the hazards.

Zhang et al. [22] propose a comprehensive hazard analysis method based on functional models. It mainly addresses fault tree analysis and FMEA.

Giese et al. [23] present an approach that supports the compositional hazard analysis of UML models described by restricted component and deployment diagrams. It also starts with environment models, but then focuses on the safety analysis of the design.

Hauge and Stølen [24] introduce the SaCS method. The method provides guidance on how to select and use patterns for the development of safety control systems. The patterns are categorized into process and product patterns. This work differs from our own, because we focus specifically on early hazard analysis and provide detailed guidance.

VII. Conclusions and Future Work

Our method has been applied in several Ford of Europe projects. However, the formal validation conditions and tool support was not used in these projects and was developed as contribution for this paper. We are confident that this contribution will ensure the same consistency and correctness of future hazard analyses with less effort than the manual approach currently used.

Our contribution has the following benefits:

- The guide-words approach avoids forgetting relevant faults and the selection of function/fault combinations beforehand helps to find the right level of detail.
- Selecting relevant situations from the hierarchically organized profile reduces the risk of forgetting a relevant situation and ensures to only consider situations that are relevant for the function in question.
- Structuring the analysis in different steps on different levels fosters an alignment between the analysis and the organizations (departments with experts regarding hardware/software, system level, vehicle/functional level) involved in the creation and review of the analysis.
- The rules for safety goal definitions help to define safety goals appropriately. Such safety goals are suitable to derive the system design.

- Our UML profile contains all relevant elements for a hazard analysis in compliance with ISO 26262. The profile provides the basis for creating and validating hazard analysis models.
- The validation conditions support the review activities required by ISO 26262.
- The validation conditions expressed as formal OCL expressions give a precise definition of the necessary checks.
- The checking of the validation conditions can be performed automatically by using our support tool.

Hazard analysis in practice is currently table-based using spreadsheets like Microsoft Excel. We are planning to work on an automatic import of these spreadsheets to our UML model. A text-to-model converter will convert entries of the spreadsheets to UML classes with the required stereotypes. The text-to-model converter will be based on the work presented in this paper. This makes it possible to apply our OCL validation conditions to the content of the spreadsheets. Hence, the spreadsheet-based hazard analysis can benefit from our approach, as well.

Acknowledgment

This research was partially supported by 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).

References

[1] UML Revision Task Force, *Object Constraint Language Specification*, http://www.omg.org/spec/OCL/2.0/PDF.

[2] M. Jackson, *Problem Frames. Analyzing and structuring software development problems*. Addison-Wesley, 2001.

[3] UML Revision Task Force, *OMG Unified Modeling Language: Superstructure*, Object Management Group (OMG), May 2010.

[4] D. Hatebur and M. Heisel, “A UML profile for requirements analysis of dependable software,” in *SAFECOMP 2010*, ser. LNCS 6351, E. Schoitsch, Ed. Springer, 2010, pp. 317–331.

[5] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), “Functional safety of electrical/electronic/programmable electronic safety-relevant systems,” ISO/IEC 61508, 2000.

[6] IEC, “Hazard and Operability Studies (HAZOP studies),” International Electrotechnical Commission (IEC), ISO/IEC 62882, 2005.

[7] DIN EN, “Analysetechniken für die Funktionsfähigkeit von Systemen – Verfahren für die Fehlzustandsart- und -auswirkungsanalyse (FMEA),” Deutsches Institut für Normung, DIN EN 60812, 2006.

[8] U. S. Department of Defense, “Electronic Reliability Design Handbook,” MIL–HDBK–338B, 1998, http://www.everyspec.com/MIL-HDBK/MIL-HDBK-0300-0499/MIL-HDBK-338B_15041/.

[9] “Road vehicles – Controller area network (CAN),” ISO 11898, 2003.

[10] International Organization for Standardization (ISO), “Road Vehicles – Functional Safety,” ISO 26262, 2011.

[11] Atos Origin, *Papyrus UML Modelling Tool*, Feb 2011, http://www.papyrusuml.org/.

[12] Eclipse Foundation, *Eclipse - An Open Development Platform*, 2011, http://www.eclipse.org/.

[13] ——, “Eclipse Modeling Framework Project (EMF),” June 2012, http://www.eclipse.org/modeling/emf/.

[14] UML Revision Task Force, *OMG Systems Modeling Language (OMG SysML)*, June 2010. [Online]. Available: http://www.omg.org/spec/SysML

[15] F. Törner, P. Johannessen, and P. Öhman, “Evaluation of Hazard Identification Methods in the Automotive Domain,” in *SAFECOMP 2006*, ser. LNCS 4166, J. Górski, Ed. Springer, 2006, pp. 237–260.

[16] S. Baumgart, “Investigations on hazard analysis techniques for safety critical product lines,” in *IRSCE’12*. New York, NY, USA: ACM, 11 2012.

[17] Safety Management System and Safety Culture Working Group (SMS WG), “Guidance on hazard identification,” Tech. Rep., 2009.

[18] P. H. Jesty, K. M. Hobley, R. Evans, and I. Kendal, “Safety analysis of vehicle-based systems,” in *Proceedings of the 8th Safety-critical Systems Symposium*, ser. LNCS 1943. Springer, 2000, pp. 90–110.

[19] Y. Papadopoulos and C. Grante, “Evolving car designs using model-based automated safety analysis and optimisation techniques,” *Journal of Systems and Software – Special issue: Computer software & applications*, vol. 76, no. 1, pp. 77 – 89, April 2005.

[20] W. Li and H. Zhang, “A software hazard analysis method for automotive control system.” IEEE Computer Society, 2011, pp. 744–748.

[21] H. Mehrpouyan, “Model-based hazard analysis of undesirable environmental and components interaction,” Master’s thesis, Linköpings Universitet, 2011.

[22] H. Zhang, W. Li, and W. Chen, “Model-based hazard analysis method on automotive programmable electronic system,” in *3rd International Conference on Biomedical Engineering and Informatics (BMEI)*, 2010, pp. 2658–2661.

[23] H. Giese, M. Tichy, and D. Schilling, “Compositional Hazard Analysis of UML Component and Deployment Models,” in *SAFECOMP*, ser. LNCS 3219, M. Heisel, P. Liggesmeyer, and S. Wittmann, Eds. Springer, 2004, pp. 166–179.

[24] A. A. Hauge and K. Stølen, “A pattern-based method for safe control systems exemplified within nuclear power production,” in *SAFECOMP*, ser. LNCS 7612. Springer, 2012, pp. 13–24.