Dipl.-Ing. Helmut Martin

Model-Based Development and Validation of Safety-Relevant E/E Systems

Dissertation

vorgelegt an der
Technischen Universität Graz

zur Erlangung des akademischen Grades
Doktor der Technischen Wissenschaften
(Dr.techn.)

durchgeführt am Institut für Elektrische Meßtechnik und Meßsignalverarbeitung
der Technischen Universität Graz
Vorstand: Univ.-Prof. Dipl.-Ing. Dr. techn. Georg Brasseur

Graz, im 11.02.2019

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am ....
(Unterschrift)

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

....
date (signature)

Auf dem Weg zu neuen Ufern ist der Sprung ins kalte Wasser nicht zu vermeiden ... das für unmöglich gehaltene ist oft nur das Unterlassene!

André Gide

Kurzfassung

Die Fahrzeugentwicklung hat eine stetig zunehmende Zahl von innovativen Funktionen zu beherrschen. Diese Funktionen betreffen heute vor allem den elektrifizierten Antriebsstrang und Fahrassistenzsysteme. Die dafür notwendige Interaktion und der Datenaustausch mit Elektrik/Elektronik (E/E) Systemen in Fahrzeugen und mit externen E/E System der Infrastruktur führen zu einer erheblichen Komplexitätssteigerung. Jegliche Art von Fehlerursachen dieser E/E Systeme können zu gefährlichen Situationen führen, besonders wenn es dabei zu Gefahren für Personen kommen kann. Wichtige Themen während der Entwicklung dieser eng verkoppelten E/E Systeme sind: (a) Einhaltung von relevanten Standards und Anleitung von Entwicklern bezüglich Qualität (z.B. AutomotiveSPICE), Funktionale Sicherheit (Safety) (e.g. ISO 26262) und auch Informationstechnische Sicherheit (Security) (z.B. SAE J3061, ISO/SAE 21434), (b) Möglichkeiten zur Wiederverwendung der entwickelten Elemente, und (c) frühe Validierung durch geeignete Simulationsmodellen.

Die vorliegende Arbeit schlägt mehrere Ansätze basierend auf Modellbasiertem Systems Engineering (MBSE) als eine Schlüsselmethode zur Beherrschung der oben genannten Herausforderungen vor. Dabei werden Modellbasierte System Safety Engineering (MBS^{2}E) Ansätze vorgestellt die folgende Bereiche adressieren: (1) Safety Prozess und Management der Arbeitsergebnisse, (2) Ausarbeitung von Safety Analysemethoden und die Ableitung von Sicherheitskonzepten basierend auf einem Systemmodell, (3) Ableitung von Simulationsmodellen für die frühe Validierung der Sicherheitsmaßnahmen und (4) Ansätze zur Wiederverwendung von verschiedenen Mustern (Pattern) für Prozesse, Design und Argumentation.

Verschiedene Anwendungsszenarien eines automobilen Traktionsbatteriesystems zeigen die Anwendbarkeit der vorgestellten Ansätze. Dabei wird die Beherrschung der steigenden Komplexität und Funktionalen Sicherheit betrachtet, der Einsatz von standardisierten Modellierungssprachen für das MBS^{2}E, und Anleitung für die Systemmodellierung und die Argumentation. Die Anwendung zeigt eine Verbesserung bei der Durchführung von Safety-Aktivitäten für die Entwicklung von sicherheitskritischen Systemen. Die relevanten Informationen für diese Aktivitäten werden dabei in einem zentralen Datenmodell konsistent abgelegt und spezifische Entwicklungstools greifen dabei auf (teilweise) wiederverwendbare Artefakte zu.

Abstract

Automotive mobility is facing a steadily increasing number of innovative functions, which are related to electrification of powertrains and advanced driver assistance systems. The required interaction and data exchange with Electric/Electronic (E/E) systems in the vehicle and with external E/E systems of the environment raises the complexity of these function. Any kind of faults and failures of these E/E systems can lead to critical hazards and can potentially harm people. A demand for the development of such interconnected E/E systems is (a) the compliance with several standards and guidance for engineers regarding quality (e.g. AutomotiveSPICE), safety (e.g. ISO 26262) and furthermore security (e.g. SAE J3061, ISO/SAE JWG 21434), (b) possibility for reuse of developed elements, and (c) early validation supported by simulation. A methodology to handle these demands (at the same time) is proposed in this thesis.

This thesis proposes different approaches based on model-based systems engineering as a key methodology to overcome the challenges stated above regarding the development of automotive E/E systems. Model-Based System Safety Engineering (MBS^{2}E) approaches are presented that supports (1) safety process and consistent work product management, (2) elaboration of safety analysis methods and derivation of safety concepts based on a system model, (3) derivation of simulation models for early safety validation, and (4) reuse of different kinds of patterns regarding process, design, and argumentation.

Different use case scenarios demonstrate the applicability of the presented approaches based on the automotive case study of a traction battery system. The use case scenarios cover handling of the increasing complexity and functional safety concerns, use of standardized modelling language for MBS^{2}E, and provide guidance for system modelling and argumentation. The application of the MBS^{2}E approaches enable an improvement of the system engineering activities by centralising the relevant information in a joint data model with interfaces to domain specific tools and reuse of dedicated safety artefacts.

Danksagung

Zu Beginn gilt mein Dank meinem Betreuer Herrn Prof. Dr. Daniel Watzenig für seine wissenschaftliche und methodische Unterstützung während der gesamten Bearbeitungsphase meiner Dissertation am Institut für Elektrische Meßtechnik und Meßsignalverarbeitung der Technischen Universität Graz.

Allen Mitarbeitern und Kollegen des Virtual Vehicle, sowie allen Beteiligten meines Studiums bin ich sehr dankbar für die stets gute und zahlreiche Unterstützung als auch die konstruktive und angenehme Zusammenarbeit in dieser Zeit. Insbesondere bedanke mich hiermit bei Herrn Martin Krammer für die gemeinsame, intensive Bearbeitung der Themen rund um das Modellbasierte Systems Engineering und Simulation.

Mein besonderer Dank gilt der Dependable Systems Gruppe rund um Herrn Dr. Christian Schwarzl, Herrn Bernhard Winkler und Herrn Robert Bramberger für die zahlreichen und unermüdlichen fachliche Gespräche, Ratschläge und Anmerkungen, die mich auf dem Weg zur fertigen Arbeit immer wieder neue Aspekte und Ansätze entdecken ließen. Auch die vielen nicht-wissenschaftlichen und motivierenden Gespräche haben dabei meine Arbeit unterstützt.

Außerdem gilt mein Dank Herrn Kurt Tschabuschnig von MAGNA Steyr Fahrzeugtechnik, der diese Arbeit aus Sicht der industriellen Problemstellungen unterstützt hat und mich bei der Bearbeitung stets durch zielführende Diskussionen begleitet hat.

Ich bedanke mich hiermit auch bei meinen Eltern, die mich auf meinem Weg durch meine Ausbildung und das Studium stets tatkräftig unterstützt und begleitet haben.

Ganz besonders möchte ich mich bei meiner lieben Frau Gabriele für Ihre unermüdliche Stärkung und Motivierung danken, wodurch auch kritische Situationen während meiner Arbeit und des Studiums gemeinsam gemeistert werden konnten. Auch meine beiden Söhne Benjamin und Niklas möchte ich an dieser Stelle dankend erwähnen, die auch sehr viel Rücksicht bewiesen haben und bei gewissen Aktivitäten auf mich verzichten mussten.

Graz, 14.01.2019 Helmut MARTIN

Extended Summary

Electric/Electronic (E/E) systems in a vehicle are safety-critical, because of possible malfunctions that could harm people. For this reason engineers have to guarantee absence of unreasonable risk for human life and the environment of E/E systems. One vital aspect in the context of the thesis at hand is to handle functional safety concerns in early design phases in an efficient way. From the functional safety perspective in the automotive domain the ISO 26262 standard provides requirements and recommendations for the entire safety life-cycle of E/E systems. Nevertheless, compliance with the standard presents a significant challenge for companies because the ISO 26262 standard only sets requirements, but does not explicitly specify how these requirements can be implemented at all. An established for quality management engineering process (e.g. AutomotiveSPICE) is needed to provide the basis for introduction of functional safety activities. This thesis discusses an integrated modelling framework of methods and tools for the development of automotive systems to cover safety-related concerns.

The Original Equipment Manufacturer (OEM) is liable for its product, that means that the product has to be safe and it must be developed according to state of the art engineering practices (standards and methods). Performing engineering activities guided and argued by cross-domain process frameworks are necessary to achieve high quality and guard OEMs against recalls. A big challenge in this context is the constraint that these engineering activities must decrease time to market and lower cost strategy of consumer products such as automotive vehicles. This requires effective engineering approaches that guide and support engineers during the development. An integrated engineering process provides a consistent basis of documentation and evidence for process compliance. Today's and future automotive mobility is facing a steady increasing number of innovative functions, which are related to electrification of powertrains and advanced driver assistance systems. The system complexity increases because of the required data exchange with E/E systems in the vehicle (e.g. sensor data for the operational strategies) and external E/E systems of the environment (e.g. for navigation in different driving scenarios). This data exchange, realised by highly interconnected E/E systems via different communication technologies, requires intelligent technology to provide their services and communicates requested data via standardized interfaces. Vehicle functions in the automotive domain are realized by software-intensive E/E systems. The role of software (SW) to implement a specific function becomes more and more important, because systems have to be flexible, configurable, maintainable and reusable. To realize such demands, standardized hardware platforms must support flexible application SW. And today more than 100 distributed Electronic Control Units (ECU) within modern vehicle are integrated because of independent development of an increasing number of functions. New technologies like multi-core processors become important, because they fulfil the demand of computational

power for the functional algorithms implemented in SW on a very small integrated design space. Furthermore, close interaction of engineering activities requires an established engineering life-cycle to handle all interactions beyond different involved disciplines such as mechanics, hardware and software.

The thesis presents the following safety-relevant topics within the Model-Based System Safety Engineering (MBS^{2}E) framework:

Safety Process and Management. A safety workflow is elaborated and relevant process elements, such as activities, tasks, roles, methods, tools, guidelines and artefacts, are modelled. The process model can be used as an input to company-specific application life-cycle management tools to coordinate process activities and required artefacts in a specific data repository. Specific safety engineering tools are used for elaboration of these artefacts. Tool interaction and traceability of safety artefacts is shown in the MBS^{2}E framework.

Safety Analysis and Concepts. ISO 26262 sets requirements for specification of safety requirements at different abstraction levels (Functional/System/Hardware/Software Level). Each level covers specific safety-related attributes supported by the MBS^{2}E approach. The work at hand covers the functional and system level of the safety life-cycle. The MBS^{2}E methodology uses a semi-formal modelling language based on SysML for modelling of safety artefacts like requirements and architectural elements, error models and safety measures. These elements are instantiated in a system model and used in an iterative approach to elaborate safety analysis artefacts by applying Hazards and Operability Analysis (HAZOP), Hazard Analysis and Risk Assessment (HARA), and Failure Mode and Effect Analysis (FMEA).

Early Safety Validation by Simulation. In this work a model-based approach for integration and test of E/E systems is proposed. The created model is used as basis for configuration of a joint simulation of different cross-domain models like thermal management, engine control and powertrain environment, and it defines the required models and their parameters to establish a link between safety goals and the structure of the simulation models. The initial model is enriched with information needed for the execution of the simulation and transformed to a language suitable for advanced simulation tasks. The execution of the simulation environment provides an early feedback to the system designers and safety engineers concerning the adequateness of the defined safety measures.

Safety Argumentation supports Safety Case. Safety argumentation is required to provide evidence that the process- and product-specific activities have been performed in a satisfying manner, that means that the management process and all decided engineering measures are adequate for the development of a safe product. The elaborated approach provides argumentation to show compliance of the engineering process in a process audit, and argumentation of safety of the product for functional safety assessment. The Goal Structuring Notation (GSN) has been used to model reusable (process and product) argumentation patterns.

Reuse of Safety Artefacts. A survey of the state-of-practice for reuse of safety artefact has been elaborated. In this survey the main industrial challenges have been identified and possible improvements have been proposed. In the thesis the following approaches for reuse are discussed: (1) Safety-oriented Process Line (SoPL) shows the possibility for reusing process elements in a company- and project-specific instantiation based on product line engineering; (2) Usage of GSN patterns show the applicability for process and product argumentation to create a Safety Case; (3) design patterns supports the elaboration of harmonized safety and security measures.

Demonstration Case Study. The applicability of the proposed MBS^{2}E approach is demonstrated in an relevant automotive use case, which provide an insight industrial engineering challenges. The automotive battery system is the major component of the powertrain of a Hybrid and Electric Vehicles. The applied MBS^{2}E approaches cover several safety concerns of the battery system and highlights modelling and traceability aspects of relevant artefacts in context of functional safety.

Handle increasing complexity and functional safety concerns. The MBS^{2}E approach provides a hierarchical abstraction and modular separation with defined interfaces between different types of modules from concept to system and sub-system level. In the context of system engineering the complex problem statement requires a adequate understanding of the nature of the underlying problem to be solved by dividing and simplifying into manageable sub-problems.

MBS^{2}E approach based on standardized modelling languages. The MBS^{2}E approach is elaborated based on system modelling using the System Modelling Language (SysML), process modelling using Software & Systems Process Engineering Meta-Model (SPEM), argumentation modelling by GSN and further extended elements to cover safety specific concerns.

Guidance for Modelling and Argumentation. The integrated MBS^{2}E approaches guide and support engineers through the relevant activities regarding design, analysis, verification by modelling of safety-relevant artefacts. Furthermore, the argumentation and elaboration of a safety case for process- and product-based argumentation is supported in a practical way.

Contents

List of Abbreviations

ADAC	Allgemeine Deutsche Automobil-Club
ADL	Architecture Description Languages
ARTEMIS	Advanced Research and Technology for EMbedded Intelligence and Systems
ASIL	Automotive Safety Integrity Level
ASPICE	Automotive SPICE
BMS	Battery Management System
BMU	Battery Management Unit
C	Controllability
COM	Communication
CPS	Cyber Physical Systems
DIA	Development Interface Agreements
E	Exposure
E/E	Electric/Electronic
E/E/PE	Electrical/Electronic/Programmable Electronic
EA	Enterprise Architect
ECU	Electronic Control Unit
ENG	Engineering
EPF	Eclipse Process Framework
EV	Electric Vehicle
FMEA	Failure Mode and Effects Analysis
FMI	Functional Mockup Interface
FMU	Functional Mockup Unit
FSC	Functional Safety Concept
FSR	Functional Safety Requirements
FTA	Fault Tree Analysis
FUN	Functional
GSN	Goal Structuring Notation
HARA	Hazard Analysis and Risk Assessment
HAZOP	HAZard and OPerability study
HEV	Hybrid Electric Vehicle
HV	High Voltage
HW	Hardware
ICOS	Independent CO-Simulation
ID	Identifier
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
JU	Joint Undertaking
MBS^{2}E	Model-Based System Safety Engineering
MBSE	Model Based Systems Engineering
MDG	Model-Driven Generation
OEM	Original Equipment Manufacterer
OMG	Object Management Group
PAA	Preliminary Architecture Assumption
PAM	Process Assessment Model
PAS	Public Available Specification
PRM	Process Reference Model
QM	Quality Managment
S	Severity
SAE	Society of Automotive Engineers
SEooC	Safety Element out of Context
SG	Safety Goal
SME	Small and Medium-sized Enterprises
SOP	Start of Production
SoPL	Safety-oriented Process Line
SOTIF	Safety of the intended functionality
SPEM	Software & Systems Process Engineering Meta-Model
SPICE	Software Process Improvement and Capability Determination
SS	Swedish Standard
STPA	Systems-Theoretic Processes Analysis
SW	Software
SYS	System
SysML	System Modelling Language
T&B	Trucks and Busses
TSC	Technical Safety Concept
TSR	Technical Safety Requirements
UML	Unified Modelling Language
WEFACT	Workflow Engine For Analysis, Certification and Test
XSTAMPP	eXtensible STAMP Platform

Chapter 1

Introduction

The development of automotive cyber-physical systems has to guarantee absence of unreasonable risk for human life. That means that such Electric/Electronic (E/E) systems in a vehicle are safety-critical, because of possible malfunctions that could harm people. For this reason engineers have to overcome several challenges: On the one hand increasing complexity due to addressing of multidisciplinary technologies, e.g. automotive battery systems for hybrid electric vehicles where electrical, mechanical, chemical and thermal disciplines play an important role [1], [2]. On the other hand the inter-connection and inter-communication functionalities of distributed systems in a vehicle and beyond vehicle's boundaries.

The main parts of this thesis consists of a collection of publications of the candidate ("Mantel" PhD thesis), that addresses approaches to improve that challenges in the context of development of safety-critical systems. The list of elaborated publications, that investigates the presented approaches are describe in a summary and in the full-text in Chapter 6.

1.1 Motivation

Today's and future automotive mobility is facing a steady increasing number of innovative functions, which are related to electrification of vehicle powertrains and advanced driver assistance systems [1]. The introduction of new functionalities requires an enhancement of established mechatronic solutions by Electric/Electronic (E/E) systems. Furthermore, these functions become more complex because of required data exchange with other E/E systems in the vehicle (e.g. sensor data for the operational strategies of the propulsion systems) and external E/E systems of the environment (e.g. for navigation in different driving scenarios). This data exchange is realised by highly interconnected E/E systems via different communication technologies and requires intelligent technology to provide their services and communicate requested data via standardized interfaces. To handle this overwhelming demand of computational power, the number of Electronic Control Units (ECU) within a car is growing.

1.2 Problem Statement

Vehicle functions in the automotive domain are realized by software-intensive E/E systems. The role of software (SW) to implement a specific function becomes more and more important, because systems have to be flexible, configurable, maintainable and reusable. To realize needed demands, standardized hardware platforms must support flexible application SW. New technologies like multi-core processors become important, because they fulfil the demand of computational power for functional algorithms implemented in SW on a very small space. Furthermore, close interaction of engineering activities require an established engineering life-cycle to handle all interactions across different involved disciplines such as mechanics, HW and SW and system integration.

1.2.1 Importance of Standards

It is vital to manage risks during the development of embedded and cyber-physical systems in safety-critical domains such as the automotive domain and other transport domains (e.g.railway, aviation). In history many recalls have been documented - several accidents with injured or even death humans occurred, where relevant safety problems could have been prevented during the product development phase:

- Airbag blew up during normal driving operation[1] (e.g. 1.2 million of the Ford model F-150 built in the years 2004 to 2006 were affected)
- Engine recalls[2] (e.g. from 2006 to 2011 worldwide 235.000 vehicles of BMW Mini cooling water pump were affected, where electric failure of the pump caused fire)
- Window lifter clamping protection[3] (e.g. in 2011 about 1.800 vehicles of Nissan model 370Z were affected, where a software failure in the window lifter occured)

In this context the life of humans may be endangered by malfunctioning on such products. This means for automotive manufacturers and suppliers of embedded systems functional safety is a top concern for their products. The automotive domain has the situation of a very large and complex supply-chain of E/E systems and that means if one major supplier may have an safety issue many other companies will be effected.

Such incidents not only seriously affect the reputation of involved companies, it could also have massive financial consequences: product recalls and legal disputes will result in huge losses.

Legal and financial consequences of standards. The functional safety standard ISO 26262 [3] is based on the notion that vehicle safety is dependent upon the behaviour of the control system itself, where it is oriented to the mass production of vehicles. The safety of a system must be validated before vehicles are put into customer market. It is highly relevant with respect to the implementation of requirements set out in European

[1] http://www.autoblog.com/2011/04/14/ford-f-150-airbag-recall-balloons-by-1-2m-units/

[2] http://www.bbc.com/news/uk-16586961

[3] http://www.nissanproblems.com/recalls/370Z/2011/

Regulation No 661/2009[4] according to which vehicle safety must be designed in line with the respective current state of science and technology. ISO 26262 provides a framework for achieving functional safety when using complex E/E systems in motor vehicles, where the standard itself is an element of the scientific and technological state of the art. Functional safety is a property of these systems which can be assessed by using the methods recommended in ISO 26262. The assessment itself reduces risks but does not completely avoid them. The standard demands for integration of its requirements into the process a quality management system based on ISO/TS 16949. The implementation of the requirements set out in the standard substantially determines the responsibilities of the manufacturers of safety-relevant systems, particularly those of vehicle manufacturers, under civil and criminal law. The ISO 26262 legal relevance of the two legal spheres of "product liability" and "producer liability" must be considered. The legal consequences of ISO 26262 do not result only from the application of a product that has been manufactured according to the processes set out in the standard. The standard itself claims to consider legal provisions and requirements defined by government authorities. The user of the standard has to comply with legal requirements in an early development phase, because the user's familiarity with requirements is a prerequisite for compliance. The standard is based on the scientific finding that absolute safety is not achievable. The right application of the ISO standard is intended to provide evidence, that a safety-related system is free from unreasonable risks. ISO 26262 can be seen as a guidance framework for processes and recommended methods for safety-relevant systems. In future, ISO 26262 on functional safety in road vehicles will be instrumental in determining the E/E architectures for automated vehicle functions. [4]

Technical impact of standards. In addition to the before mentioned reasons of potentially protecting human lives, legal and business interests of automotive industries. There is another reason to implement adequate risk management processes and safety measures. The automotive domain is a highly regulated sector, with various standards stipulating the relevant safety requirements. Failing to prove compliance with relevant development standards means that the company will not have the possibility to enter the automotive market.

International standards such as IEC 61508 [5] or ISO 26262 can be seen as fundamental in automotive sector. Regarding functional safety (safety of the E/E system), the IEC 61508 is the basic international functional safety standard applicable to electrical/electronic/programmable electronic (E/E/PE) safety-related systems of all industries. ISO 26262 for road vehicles' functional safety is an adaptation of this standard that is applicable to the development of safety-related E/E systems in the automotive domain. One important aspect of functional safety is the risk of electronic malfunction, e.g malfunction of the battery control unit caused by incorrect inputs or software errors. These malfunctions could lead to hazardous events for passengers, other traffic participants, and uninvolved parties (e.g. fire due overcharging of battery cells). The risk of malfunctions has to be lowered to an insignificant risk potential by gaining a clear understanding of possible faults, as well as their causes and effects, and by providing solutions for fault mitigation. One vital aspect in the context of the thesis at hand is to handle functional safety concerns in early design phases in an efficient way. Additionally, due to the close interac-

[4] Official Journal of the European Union dated 31 July 2009, L 200/1

tions within mechanics, electronics and software, the analysis and handling of transversal intra-domain effects are gaining more importance.

From the functional safety perspective in the automotive domain the ISO 26262 standard provides requirements and recommendations for the entire safety life-cycle of E/E systems. That means this ISO standard covers the question:

- *WHAT has to be done to provide evidences for compliance with functional safety requirements according to ISO 26262?*

The main challenge is that ISO 26262 doesn't provide a clear guidance for engineers to support the introduction of functional safety in companies and following questions arise:

- *HOW should the standard be applied in a company in an economic an efficient way?*
- *HOW can a company argue and provide "enough" evidence that ISO 26262 is fulfilled and the system is safe?*

Quality Standards. The safety management according ISO 26262 demands an established quality engineering process in the organization and during specific product development projects. An existing evidence related to a quality management system complying with a quality management standard, such as ISO/TS 16949 or equivalent is needed.

In the automotive sector processes for software-intensive embedded systems can be described by Automotive SPICE (ASPICE) [6] (customization derived from the initial ISO/IEC 12207) or derived from ISO/TS 16949 requirements (customization derived from the more general ISO 9001 standard). The quality assurance group in a company works on achieving the compliance with the ISO/TS 16949 standard for certification purposes of the overall quality business process. On the other side a technical, more focused, group works on best practices in ASPICE to be applied for process improvement purposes regarding embedded system development. In this thesis ASPICE for quality assurance will be covered for the quality aspects. In summary, an established engineering process is needed to provide the basis for introduction of any standard compliant quality and functional safety activities.

1.2.2 From document centric to model-based standard compliant work products

For all engineering activities, specific documentation is required to prove that the development of a product conforms to the standard. ISO 26262 requires specific evidences by work products, which are related to requirements of the standard for each safety activity. However, different work products share the same safety artifact data. Thus, if any of these artifacts have to be changed, different work products are affected and have to be updated to provide consistent information. This is the main drawback of document-centric development. The introduction of semi-formal notation (e.g. UML, SysML) improves this situation because the artifacts in a system model use a consistent and shared data source. The required work products can therefore be created and exported as the documentation output of the system modelling effort. The system modelling described in this thesis is based on the international standard Object Management Group (OMG) Systems Modeling Language (SysML) [7], since this language has been successfully applied and is supported by commercial available modelling tools. A standardized, semi-formal modelling language paves the way from the document-centric to the model-centric development approach.

Safety Abstraction Levels. ISO 26262 defines a safety lifecycle, which is oriented toward the general V-Model [8] of product development and by delineating different levels of safety requirements (see Figure 1.1). The safety issues are thereby outlined from a higher level of functional abstraction (safety goal, functional safety requirements) to the more detailed levels of the technical realization of a system (technical safety requirements), down to the software (software safety requirements) and hardware levels (hardware safety requirements). The following paragraphs provides a separation between problem and solution space in the context of ISO 26262.

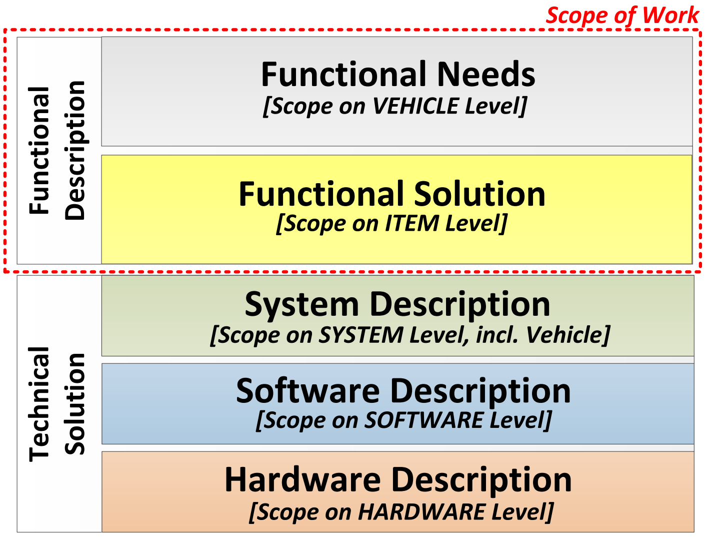


Figure 1.1: Overview of abstraction levels (main scope on thesis on functional level)

Functional Needs cover the definition of the item[5]. The boundaries of the item and the interaction of the item with different stakeholders (e.g. other items, other technologies, users/ humans) has to be defined. Based on this "item definition", the Hazard Analysis and Risk Assessment (HARA) analyzes any potential safety-risks. The HARA determines an Automotive Safety Integrity Level (ASIL) for risk classification and a number of safety goals. Safety goals represent top-level safety requirements for subsequent safety activities in the safety lifecycle.

Functional Solution of the item has to be defined independently of any technical solution. This separation is crucial for the understanding of the characteristics and in-

[5] An item represents a system or an array of systems that realizes a specific functionality on the vehicle level (ISO 26262 part 3).

teractions of such complex automotive systems. The first level of the solution is covered in the functional safety concept (FSC) (ISO 26262 part 3), which is derived from the safety goals. The FSC defines all required safety measures for handling different kinds of identified malfunctions. These safety measures have to be reflected as Functional Safety Requirements (FSR), which must be allocated to the elements of the item. The main focus of this thesis is on the above presented functional description, which covers functional needs and functional solutions. This a vital part of the safety lifecycle because the ASIL and the FSC provide the basis for the technical solution and all subsequent safety activities.

Technical Solution of the item defines the technical implementation of system including hardware and software. The system description (ISO 26262 part 4) defines the Technical Safety Concept (TSC), which refine the FSC by specifying technical safety mechanisms. Safety mechanisms cover detection, indication and control of faults, both in the system and in external devices that interact with the system. Safety analyses of the system design are needed to avoid systematic failures, random HW failures, and common cause failures, as well as to verify the TSC. On the lowest level of the system implementation, the TSC has to be refined by HW (ISO 26262 part 5) and SW (ISO 26262 part 6) components.

1.2.3 Simulation supports Early Safety Validation

Following directly the approach of ISO 26262 some limitations can be observed in traditional engineering approaches. First, a lack of traceability between safety goals, their derived safety requirements, created work products, developed components, and the resulting system. This complicates the argumentation of a product's safety. Second, the automotive supply chain must be aligned to support these safety activities. Horizontal integration issues (e.g. different software-components) and vertical integration issues (e.g. component, module and system levels) as well as testing problems arise in this context. Integration test cases applied during document-centric approaches might not be as correct and complete as they seem, due to missing links to the initial safety goals. To overcome these issues, this thesis proposes a combined solution covering two main aspects, namely system modeling and virtual prototyping. For a formalized system description approach, we decided to deploy SysML and SystemC languages. These languages complement each other and provide the possibility for model-centric development. They support the need for the characterization of structure, behavior, requirements and simulation of automotive embedded systems. The resulting models are used as a reference throughout requirements and system design phases and serve as a basis for component design and implementation phases. Different views on the model are used to extract necessary information for integration and test scenarios, which span across horizontal and vertical levels. The desired functions are rooted in different engineering domains. Therefore a co-simulation approach is followed: By coupling different simulation tools, the impact and interaction of HW, SW, and mechatronic/mechanical components on the vehicle's functions can be observed. That approach provides the possibility to perform fault injection to expose safety goal violations in early development phases. In context of functional safety, the created models and corresponding test scenarios are oriented at the overall safety goals and form an executable safety case, providing arguments for safety validation. The introduced concepts and methods are demonstrated by an automotive use case.

1.2.4 Safety Assessment/Safety Case

Companies, which deal with safety critical products, engage external authorization bodies to certify their abilities concerning functional safety development (e.g. functional safety audit and functional safety assessment). Safety certification ensures that a certain product fulfills specific safety requirements in a specific environment. It requires a complete and structured collection of evidence to show that the developed system is acceptably safe. The role of safety arguments is often neglected, thus stakeholders who are not directly involved in the creation of work products (e.g. reviewers) may have troubles to reconstruct the train of thought concerning decisions taken. Documentation of decisions in a comprehensible manner avoids loss of crucial information. A systematic approach is required to handle the development process that deals with dependency issues of the elaborated work products, because the complex relationship between them may be not obvious. Standard compliant work products represent artifacts, that cover outcomes of a specific engineering task. An argumentation method is needed that accompanies the process and is able to deal with the complex linkage between these individual artifacts. In order to come up with a versatile approach, being capable of dealing with a broad range of complex systems and processes, this method must be structured, modular and scalable. A vital topic in context of ISO 26262 is the elaboration of the safety case. It defines a safety case as "the compilation of all work products that are used as evidence to show that all requirements for an item are satisfied. [...] The three principal elements are requirements, arguments and evidence". Arguments explain the relationship between evidence and requirements, but ISO 26262 does not provide detailed requirements concerning creation of the safety cases.

1.2.5 No Safety without Security

As mentioned in the beginning, the rising vehicle connectivity (vehicle-to-vehicle, vehicle-to-infrastructure) causes multiple inter-vehicle connections as well as capabilities for (wireless) networking with other vehicles and non-vehicle entities. Connections are not restricted to internal systems (e.g. steering, sensor, actuator, and communications) but also include other road users and infrastructure. Current vehicles already utilize connectivity for over-the-air updates, smart maintenance, remote tracking or insurance services. A well-known demonstration of security risks was the hack of a Jeep Cherokee [9]. The intrusion started through a vulnerability in the cellular network configuration, progressed from the telematic system and ultimately affected even safety-critical control units. The Attackers were able to influence braking, steering and acceleration. A similar weakness was also found by the German automotive club ADAC in the ConnectedDrive system installed in BMW vehicles. A vulnerability in the communication configuration allowed an attacker to access the communication. [...] While reliability and functional safety are well accepted in the automotive domain, security engineering is a novel aspect for this industry. Even if the hand-over of know-how and best practice from other application domains can be performed successfully, a major aspect here is the efficient integration of reliability, safety and security engineering into a common lifecycle for the development of dependable automotive cyber-physical systems [10]. This work investigates an co-engineering approach that extends the proposed methodologies for security aspects regarding process modelling and argumentation.

1.2.6 Research Goals and Methods

The following research goals are investigated in this thesis:
Research Goal 1: Definition of a model-based development method ensuring compliance to safety and quality standards.
Research Goal 2: Development of model-based system development approach maintaining traceability and consistency between work products and allows for early safety validation.
Research Goal 3: Development of a method for creating a safety argument considering security aspects.

Different research methods have been investigated for the elaboration of the thesis: Investigation of industrial practices and literature research especially on norms and standards regarding functional safety and systematic reuse. Development of model-based methodology in parallel to existing, document based approaches to show the benefits of MBSE approaches. For the demonstration part experimental evaluations have been performed based on industry sized problems within industry driven research projects to show the applicability of the approaches. Furthermore, advantages and challenges of the proposed methods have been investigated and novel research topics have been identified.

1.2.7 Deliminations

Focus in this thesis is on the concept phase, because identification of any kind of malfunctions in this phase is most beneficial and can vastly reduce development cost. The presented results can form the basis for well-established development processes for technical implementation like entire systems, hardware or software. It supports the development process by ensuring consistency and traceability as well as early validation using simulation (e.g. co-simulation).

The field of safety-engineering is quite broad and so the thesis focuses a specific area as described above and only references to the following aspects of the development process:

- Technical implementation of system design by hardware and software;
- Integration, verification and validation aspects of technical implementation; and
- Life-cycle phases production, operation, maintenance and decommissioning.

1.3 Organisation of the Thesis

The remainder of the thesis is organised as follows: Chapter 2 describes related work in the areas automotive safety engineering regarding process and system modelling, safety analysis, and safety argumentation. The Chapter 3 presents the main methodology approaches of process modelling, model-based safety engineering, safety argumentation and co-engineering framework. In Chapter 4 the automotive case study high voltage battery system of an hybrid electric vehicle powertrain is used for demonstrating the applicability and experimental evaluation of the elaborated methodologies on specific use case scenarios. Chapter 5, provides a conclusion of the overall work and gives an outlook on possible next steps and further upcoming challenges in the development of safety-critical automotive embedded systems. The last Chapter 6 provides a summary of elaborated and relevant publications, includes all relevant publications of the author and an allocation of the papers to the specific contributions of the thesis.

1.4 Involved Projects

The following European research projects provided a valuable contribution

1.4.1 SafeCer Project

The project SafeCer (Safety Certification of Software-Intensive Systems with Reusable Components) is an international research collaboration targeting increased efficiency and reduced time-to-market by the composable certification of safety-relevant embedded systems. The project was separated in two sub-project namely pSafeCer and nSafeCer.

pSafeCer Project. The two-year pSafeCer (pilotSafeCer) [11] project was started in April 2011 and is funded partly by the ARTEMIS Joint Undertaking (the European Public-Private Partnership for Advanced Research and Technology for EMbedded Intelligence and Systems) and partly by national funding. pSafeCer aims to support arguments for the reuse of safety certification and pre-qualified components within and across industrial domains. This addresses the overarching goal of the ARTEMIS JU strategy to overcome fragmentation in the embedded systems markets so as to increase the efficiency of technological development while facilitate the establishment of a competitive market in the supply of embedded systems technologies and provide market access for SMEs as suppliers of trustworthy (pre-qualified) components, qualified tools and software (meeting specific SME related target of ARTEMIS Innovation Environment). pSafeCer applies mainly existing/adapted methods and techniques by integrating them via interfaces and transformations, thereby clearly meeting the ARTEMIS JU over-arching objective of closing the design productivity gap between potential and capability.

nSafeCer Project. European industry has a great potential to achieve a leading position in the growing global market of safety-relevant embedded systems, provided it can devise efficient and industrial-strength methods and processes for their development and certification. The three-year nSafeCer (novelSafeCer) [12] project targeted increased efficiency and reduced time-to-market by composable safety certification of safety-relevant embedded systems. nSafeCer built on the ARTEMIS pilot project pSafeCer. Sharing the same overall goals, the concepts developed in pSafeCer were advanced into tangible industrial implementations of “project-ready“, unified and seamlessly integrated solutions, and demonstrators of the proof of concepts. nSafeCer aimed to support arguments for reuse of safety certification and pre-qualified components within and across industrial domains. This addressed the overarching goal of the ARTEMIS JU strategy to overcome fragmentation in the embedded systems markets so as to increase the efficiency of technological development. Furthermore, it facilitated the establishment of a competitive market in the supply of embedded systems technologies and market access for SMEs. nSafeCer adds scientific objectives, including support for product lines and cross-domain certification and reuse.

The whole SafeCer project, which includes pSafeCer and nSafeCer projects contribute the following to the thesis:

- Definition of the hybrid powertrain use case and relevant engineering scenarios
- Semiformal process modelling and reuse of process elements
- Investigation on model-based systems engineering and interaction with safety analysis activities
- Introduction to the topic of safety argumentation

1.4.2 VeTeSS Project

VeTeSS (Verification and Testing to Support Functional Safety Standards) [13] developed standardised tools and methods to verify the safety of automotive embedded systems. By providing objective data for safety qualification and certification, reliance on manual processes and expert opinion were reduced as well as development costs and time to market, even with the increasing complexity of embedded systems and software. The project defined a seamless, automated design flow from requirements to validation, integrating formal analysis, simulation and physical test. Based on this, a set of consistent tools and methods for safety analysis and testing across HW and SW, and analogue and digital domains have been provided. This included methods and tools to automatically derive test procedures and test vectors to validate an architecture against the safety goals.

The VeTeSS project contributes the elaboration of model-based simulation frameworks to the thesis.

1.4.3 EMC2 Project

EMC2 (Embedded Multi-Core systems for Mixed Criticality applications in dynamic and changeable real-time environments) [14] found solutions for dynamic adaptability in open systems. It provided handling of mixed criticality multi-core applications in real-time conditions, with scalability and utmost flexibility, fullscale deployment and management of integrated tool chains, through the entire lifecycle. The EMC2 project focused on the industrialisation of European research outcomes and builds on the results of several previous ARTEMIS, European and national projects. It provided the paradigm shift to a new and sustainable system architecture that was capable of handling open dynamic systems. EMC2 expected to facilitate the EU's ability to deploy and use embedded systems across important European market sectors (e.g. Automotive Embedded Systems as a key innovation driver, enabling the majority of innovations).

The EMC2 project contributes by:

- Elaboration of process framework including safety argumentation
- Investigation in safety and security co-engineering

1.4.4 AMASS Project

AMASS (Architecture-driven, Multi-concern and Seamless Assurance and Certification of Cyber-Physical Systems) [15] AMASS project will create and consolidate the de-facto European-wide open tool platform, ecosystem, and self-sustainable community for assurance and certification of Cyber-Physical Systems (CPS) in the largest industrial vertical markets including automotive, railway, aerospace, space, energy. The ultimate goal of AMASS is to lower certification costs for CPS in face of rapidly changing features and market needs. This will be achieved by establishing a novel holistic and reuse-oriented approach for architecture-driven assurance (fully compatible with standards such as AUTOSAR and IMA), multi-concern assurance (for co-analysis and co-assurance of e.g. security and safety aspects), and for seamless interoperability between assurance and engineering activities along with third-party activities (e.g. external assessments and supplier assurance).

The AMASS project contributes by:

- Investigation in safety and security co-engineering
- Pattern-based approach for safety and security measures

Chapter 2

Related Work and Background

2.1 Relevance of Automotive Standards

The main relevant standards for this thesis are Automotive SPICE (ASPICE) and the automotive functional safety standard ISO 26262, which are described in the following.

ASPICE - "Automotive SPICE Process Assessment\Reference Model v3.0" covers two main aspects (1) ASPICE Process Assessment Model (PAM) and (2) ASPICE Process Reference Model (PRM), which are incorporated since v3.0 and they are used in conjunction when performing an process assessment. *"The ASPICE PAM is intended for use when performing conformant assessments of the process capability on the development of embedded automotive systems. The PAM offers indicators in order to identify whether the process outcomes and the process attribute outcomes are present or absent in the instantiated processes of projects and organizational units. Process performance indicators, which apply exclusively to capability Level 1. They provide an indication of the extent of fulfillment of the process outcomes. Process capability indicators, which apply to Capability Levels 2 to 5. They provide an indication of the extent of fulfillment of the process attribute achievements. BPs represent activity-oriented indicators. WPs represent result-oriented indicators. Futhermore, the PAM offers a set of work product characteristics for each WP. [..] The ASPICE PRM was developed based on the former version ASPICE PRM 4.5 and it was further developed and tailored considering the specific needs of the automotive Quality Management System. If processes beyond the scope of ASPICE are needed, appropriate processes from other PRM such as ISO/IEC 12207 or ISO/IEC 15288 may be added based on the business needs of the organization."* [6]
In this work, the process performance indicators of BP and WP are relevant to cover the outcomes of the base development process.

ISO 26262 - "Road Vehicles - Functional Safety" [3] is the automotive industry-specific derivation of the generic industrial functional safety standard IEC 61508 [5]. The first edition of ISO 26262 was released in November 2011 as the state of the art international standard for E/E systems in series production passenger cars with a maximum gross weight of 3500kg. The standard relates to the functional safety of E/E systems, not to that of systems as a whole or of their mechanical subsystems. These safety-relevant processes may be viewed as being integrated or running in parallel with a managed requirements lifecycle of a conventional Quality Management System (e.g. oriented on ASPICE). ISO 26262 provides a structured and generic approach for the complete safety lifecycle of an automotive E/E system, including design, development, production, service processes and decommissioning. It defines Automotive Safety Integrity Level (ASIL) as a risk classification parameter for the safety-critical hazardous situations of an item[1]. This is an important parameter for all subsequent safety activities within safety lifecycle. The ASIL can be seen as a parameter that indicates the amount of effort for risk reduction in order to achieve a tolerable risk level. After the determination of the ASIL it is used for different safety activities over the whole standard to provide a recommendation levels for best-practice methods based on the ASIL.

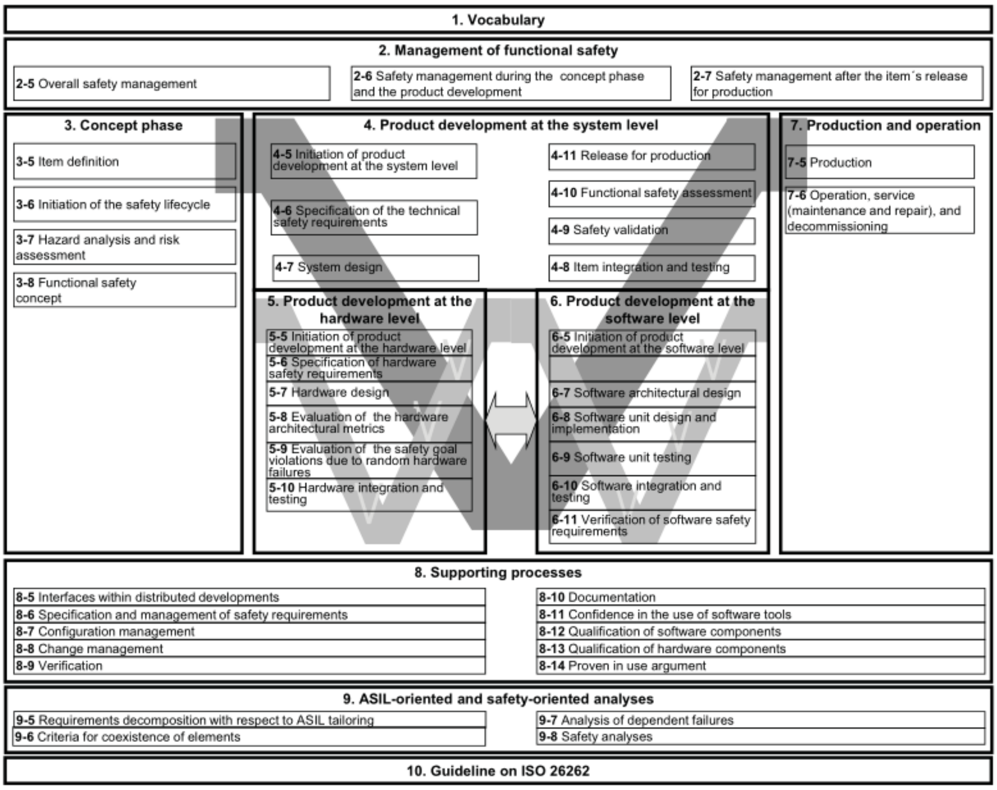

1. Vocabulary
2. Management of functional safety
2-5 Overall safety management
2-6 Safety management during the concept phase and the product development
2-7 Safety management after the item´s release for production
3. Concept phase
3-5 Item definition
3-6 Initiation of the safety lifecycle
3-7 Hazard analysis and risk assessment
3-8 Functional safety concept
4. Product development at the system level
4-5 Initiation of product development at the system level
4-6 Specification of the technical safety requirements
4-7 System design
4-11 Release for production
4-10 Functional safety assessment
4-9 Safety validation
4-8 Item integration and testing
7. Production and operation
7-5 Production
7-6 Operation, service (maintenance and repair), and decommissioning
5. Product development at the hardware level
5-5 Initiation of product development at the hardware level
5-6 Specification of hardware safety requirements
5-7 Hardware design
5-8 Evaluation of the hardware architectural metrics
5-9 Evaluation of the safety goal violations due to random hardware failures
5-10 Hardware integration and testing
6. Product development at the software level
6-5 Initiation of product development at the software level
6-7 Software architectural design
6-8 Software unit design and implementation
6-9 Software unit testing
6-10 Software integration and testing
6-11 Verification of software safety requirements
8. Supporting processes
8-5 Interfaces within distributed developments
8-6 Specification and management of safety requirements
8-7 Configuration management
8-8 Change management
8-9 Verification
8-10 Documentation
8-11 Confidence in the use of software tools
8-12 Qualification of software components
8-13 Qualification of hardware components
8-14 Proven in use argument
9. ASIL-oriented and safety-oriented analyses
9-5 Requirements decomposition with respect to ASIL tailoring
9-6 Criteria for coexistence of elements
9-7 Analysis of dependent failures
9-8 Safety analyses
10. Guideline on ISO 26262

Figure 2.1: Overview of different parts of ISO 26262 [3]

The ISO 26262 standard consists of 9 normative parts and a informative guideline as the 10th part (see figure 2.1). The most relevant parts of ISO 26262 for this thesis are part 1, 2, 3, 4, 8, 9, which are underlined in the following section.
The ten parts of ISO 26262 are:

- **Part 1 - Vocabulary:** Terminology and glossary of relevant terms within the standard.
- **Part 2 - Management of functional safety:** Specifies the requirements regard-

[1] An item is a system or array of systems for implementing a function at vehicle level, to which ISO 26262 is applied. [3]

ing functional safety management for automotive applications, including project-independent requirements (organizational activities, i.e. overall safety management), project-specific requirements (management activities in the safety lifecycle, i.e. management during the concept phase and product development, and after the release for production).

- **Part 3 - Concept phase:** Defines the requirements for the concept phase for automotive applications, including the item definition, initiation of the safety lifecycle, hazard analysis and risk assessment and functional safety concept.

- **Part 4 - Product development at the system level:** Covers requirements of the left leg of the V-Model for system design including initiation of product development at system level, specification of technical safety requirements, technical safety concept, system design. Furthermore, it covers the right leg of the V-Model for verification and validation including item integration and testing, safety validation, functional safety assessment, and release for production.

- **Part 5 - Product development at the hardware level:** Specifies the requirements for automotive applications, including requirements for initiation of product development at the hardware level, specification of the hardware safety requirements, hardware design, hardware architectural metrics, and evaluation of violation of the safety goal due to random hardware failures and hardware integration and testing.

- **Part 6 - Product development at the software level:** Defines requirements for initiation of product development at the software level, specification of the software safety requirements, software architectural design, software unit design and implementation, software unit testing, software integration and testing, and verification of software safety requirements.

- **Part 7 - Production and operation:** Specifies the requirements for production, operation, service and decommissioning.

- **Part 8 - Supporting processes:** Includes interfaces within distributed developments, overall management of safety requirements, configuration management, change management, verification, documentation, confidence in the use of software tools, qualification of software components, qualification of hardware components, and proven in use argument.

- **Part 9 - ASIL-oriented and safety-oriented analysis:** Covers requirements decomposition with respect to ASIL tailoring, criteria for coexistence of elements, analysis of dependent failures, and safety analyses.

- **Part 10 - Guideline on ISO 26262 (informative):** This part provides an overview of ISO 26262, as well as giving additional explanations, and is intended to enhance the understanding of the other parts of ISO 26262. It describes the general concepts of ISO 26262 in order to facilitate comprehension. The explanation expands from general concepts to specific contents.

Combination of Quality and Safety. ISO 26262 prescribes both functional safety assessments considering the product being developed and functional safety audits considering the development process for this product. However, ISO 26262 provides limited guidance on how to perform these functional safety assessments and audits. Different activities have been investigated in integration of ASPICE and Functional Safety. Messnarz et al. [16, 17] discusses in these papers how the functional and requirements traceability concepts in ASPICE have to be extended to cover the criteria and content demanded by ISO 26262. Furthermore, the papers describe how these new concepts are considered in the integrated ASPICE and Safety assessment approach which was proposed. The Swedish Standard SS 7740 is a response to the needs of such guidance that could be commonly used in the automotive industry [18]. The main purpose of the SS 7740 assessment model is to standardize assessments of functional safety processes including well-defined capability levels, i.e. ISO 26262 functional safety audits with standardized capability levels. SS 7740 complements an ordinary ASPICE assessment with respect to ISO 26262 by extending the ASPICE PAM with a set of ISO 26262 unique indicators. These indicators may also be used when implementing a process improvement program following an assessment, or as a means to guide a functional safety assessment that is focused on practices and the quality of work products.

2.2 Modelling and Analysis of Safety Artefacts

2.2.1 Process Modelling

Process Modelling Languages. In the literature, several process modelling languages are available [19], [20], [21]. SPEM (Software Process Engineering Meta-model)2.0 [22] is the OMG (Object Management Group)'s standard for software process modelling. Furthermore, SPEM 2.0 is simply one of them but since it has appealing features in terms of standardization, reuse, tool-support, etc. (as surveyed in [21]) as well as in terms of active community working towards its enhancement [23], it answers the expectations for the thesis. SPEM 2.0 offers static as well as dynamic modelling capabilities, the latter achieved by including links to other modelling languages (e.g. UML activity diagrams). SPEM 2.0 also offers modelling capabilities to address process variability. As explored in [24] these modelling capabilities are not fully adequate to model process lines. However, the alternative modelling proposal [25], called vSPEM, which is currently matter of investigation, is still too immature to be considered in the time-frame of our project.

Safety-oriented Process Line. A process line [26] is a family of highly related processes that are built from a set of core process assets in a pre-established fashion. A safety-oriented process line is a process line that targets safety processes [24]. A (safety-oriented) process line approach is constituted of three phases: scoping (i.e. definition of the set of processes to be examined as a family), domain engineering (i.e. commonality and variability identification and modelling), process engineering (modelling of single processes via selection and composition of reusable commonalities and variabilities). Comparisons among safety processes characterize the main activity of the domain engineering phase. Through comparisons, it is possible to identify what can vary (variabilities) between safety processes and what, remains unchanged (commonalities). At a first glance, processes defined in different standards seem to exhibit only variabilities. Terminological differences constitute a barrier to a straightforward identification of commonalities.

2.2.2 System Modelling

System Modelling Language. The work of Dajusuren et al. [27] discussed a number of Architecture Description Languages (ADLs) (SysML, EAST-ADL, AADL, TADL, AML, and MARTE) and evaluated a set of ADLs based on the automotive-specific modeling requirements. They selected SysML as a viable language to carry out the case study on automotive systems and to demonstrate a method for architectural consistency checking using SysML. The use of the SysML diagram types was evaluated, and the benefits and disadvantages of the features were discussed from the perspective of the automotive domain. SysML is a general purpose modeling language. It is based on Unified Modeling Language (UML), and was constructed for systems engineering applications. It is standardized by the OMG[2] [7]. The SysML concepts concern requirements, structural modeling, and behavioral constructs. New diagrams include a requirement diagram and a parametric diagram and adjustments of UML activity, class, and composite structure diagrams. Tabular representations of requirements or allocations, for example, are also included as an alternative notation. The three main diagram types of SysML are requirement diagram, block definition diagrams and internal block diagrams. The requirement diagram provides cross cutting relationships between requirements and system models; the structural diagrams are block definition diagrams, internal block diagrams, package diagrams, and parametric diagram; the behavioral diagrams are use case, state machine, activity diagrams, and sequence diagrams. SysML requirement diagram provides the relationship types satisfy, verify and trace, which enable requirements traceability. Any SysML modeling element can be connected to the requirement via trace relationship to enable a traceability, which is considered weak as its semantics do not include any constraints [28]. In contrast to SysML, the modelling languages ADL, AML, and MARTE do not support explicitly the requirements traceability, which is a vital topic for the development of safety-critical products according the ISO 26262.

Safety Analysis Methods. The following qualitative safety analysis methods are recommended in the ISO 26262 for hazard identification, derivation of safety requirements, system/hardware/software design analysis, validation at the vehicle level. The HAZOP is an analysis method for identifying potential safety and operational problems associated with design, maintenance or operation of a system. A HAZOP is a formal and objective process, which ensures a systematic and well-documented evaluation of potential problems/hazards [29, 30].

The HARA is a safety analysis methodology for the automotive domain, which provides a systematic determination of potential risks in specific driving situations. The HARA introduces the ASIL for each hazardous event and defines Safety Goals as high level safety requirements (ISO 26262 [3]-Part 3).

The Failure Mode and Effects Analysis (FMEA) is a systematic method to analyze potential failure modes aimed at preventing failures. It is intended to be a preventive and

[2] http://www.omg.org/

detection action process carried out before implementing new features or changes in products or processes [29–31].

Combination of Safety Analysis and System Modelling. In [32] an FMEA analysis was conducted based on two different diagram types of SysML. An interesting approach is also discussed in [33], where a SysML modelling process for software, electronics and mechanics is introduced. Various safety aspects are targeted and a preliminary hazard analysis is conducted. Piques et al. described their industrial experiences in applying a SysCARS (System Core Analyses for Robustness and Safety) methodology in industrial projects [34, 35]. The SysCARS methodology provides a precise mapping of system engineering artifacts to SysML artifacts, as well as the sequence of modelling of activities to be performed, by a "workflow-driven" mechanism. Moreover, they show how interoperability is ensured with the tools already in place for requirements management and control design. The works of Mader et al. [36–39] describe an safety engineering approach called OASIS (AutOmotive Analysis and Safety EngIneering InStrument) for combining system modelling by using EAST-ADL and safety analysis methods. They demonstrated the applicability of OASIS by using an open source tool-chain based on Eclipse.

2.3 Early System Validation by Simulation

Virtual Validation. In future simulation plays an increasing role in verification and validation of modern cars because of its advantage in varying the virtual environment easily and representing different variations of cars. These tests can be monitored and reproduced at any time. Another advantage of simulation is that it can be run day and night and furthermore massively in parallel if needed [40]. The validation by simulation is a vital topic in the development of complex Cyber Physical Systems (CPS), for both industry and academia. In research, complex questions of combining different computation models for coupled (co-) simulation are of specific interest. Platforms and simulators are integrated to reveal effects and hidden interactions that would not show up if only a single aspect would be analyzed. ISO 26262 recommends simulation explicitly as a quality assurance technique for verification of system design artifacts [41].

Simulation Language System-C. SystemC [3] is a C++ based library for modelling and simulation purposes. It is intended for the development of complex electric and electronic systems [42]. SystemC targets high abstraction level modelling for fast simulation [43]. It provides sets of macros and functions, and supports paradigm like synchronization, parallelisms, as well as inter-process-communications. Its simulation engine is included in the library, and is built into an executable during model compilation. While SystemC is capable of modelling and simulating digital systems, its SystemC-AMS[4] extension expands these concepts to the analog and mixed signal domain. Both, SystemC and SystemC-AMS libraries, provide a certain degree of protection of intellectual property, when optimized and compiled models are exchanged. SystemC and SystemC-AMS are used in a variety of simulation platforms, where different wrappers or adapters provide

[3] http://www.accellera.org

[4] http://www.systemc-ams.org

data exchange services. Examples thereof are given in [44], [45]. Regarding the use of SystemC or SystemC-AMS in the context of the FMI standard, no relevant publications are available to date, describing a unified process for integration.

Co-Simulation Standard. The Functional Mockup Interface[5] (FMI) is an open standard, which defines an interface supporting model exchange between simulation tools and interconnection of simulation tools and environments. The second version of the FMI standard was released in 2014 [46], [47].

The FMI is introduced and argued about the necessity to share models for model/ software/ hardware-in-the-loop testing activities [48]. As part of that a methodology for gradual integration and progressive validation is proposed. It also emphasises the need for conversion of existing models into the FMI standard.

Chen discusses technical issues and implementation of a generic interface to support the import of functional mock-up units into a simulator [49]. For this import, the FMI calling sequence of interface functions from the standard are used.

Noll describes the implementation of FMI in SimulationX. It presents code generation out of a simulation model for Functional Mockup Units (FMUs) for model exchange and co-simulation [50]. A code export step generates the necessary C-code for model exchange. For co-simulation, a solver is included in the resulting Dynamic Link Library. The tool coupling using SimulationX is accomplished by using a wrapper.

The need for co-simulation in connection with the design of cyber-physical systems is highlighted in [51]. It follows the idea, that coded solvers in FMUs have some limitations regarding analysis or optimization. Therefore the authors strive for explicitly modelled ordinary differential equation solvers and claim a significant performance gain.

In [52] the generation of FMUs from software specifications for cyber-physical systems is outlined. This approach fulfills the need for software simulation models. A UML based software specification is automatically translated into a FMU, maintaining its original intended semantics. This step is done using C-code, which is included within the FMU.

In [53] a high level approach for integration and management of simulation models for cyber-physical systems is shown.

An integration strategy for rapid prototyping for Modelica models into the FMI standard is presented in [54]. It highlights a high level approach for integration of cyber-physical systems.

2.4 Safety Case - Safety Argumentation

Safety Argumentation and ISO 26262. A very important topic is the elaboration of a safety case in context of ISO 26262, but there are no detailed requirements in the standard concerning the elaboration of the safety cases. The standard defines a safety case as the compilation of all work products that are used as evidence to show that all requirements for an item are satisfied. *[...]* The three principal elements are requirements, arguments and evidence. Arguments explain the relationship between evidence and requirements (objectives). Distributed development is omnipresent in the automotive domain. ISO 26262 defines Development Interface Agreements (DIA) for clarification of

[5] http://www.fmi-standard.org

the relationship between OEM and different suppliers (Tier x). DIA connects safety cases, if distributed development is performed. If we have a look to other domains, it can be seen that safety cases are regarded as important and that they obtain a lot of attention. Depending on the context different stages of safety cases can be defined. The British "Office for Nuclear Regulation" [55] defines 11 principal stages in the life cycle of a nuclear facility. Kelly [56] defines three software safety cases based on the "MoD Defence Standard 00-55" [57] from the military domain.

Goal Structuring Notation (GSN) is a graphical notation that can be used to document safety arguments in a safety case [58]. The timely generation of well-focused safety cases is capable of bringing considerable benefit in the context of development and assessment. The safety assurance of automotive E/E systems according ISO 26262 shall cover process and product aspects. A process-based argumentation only renders the standard's implicit argumentation in a different form. Further argumentation is needed to provide a rationale for product-specific decisions during the development [59]. A process argumentation approach to generate process-based arguments from process models is shown in [60]. It reduces cost and time during the certification process. Distinction between process- and product-based argumentation has been made in [61] but only product-based argumentation has been considered in detail. It deals with building of reusable safety cases and patterns. The authors in [62] propose integrated process- and product- based argumentation. Process-based arguments are backing arguments for product-based arguments to derive the safety case. The safety case development manual [63] provides guidance on the development of safety cases for the avionic domain. In this manual a clear distinction between product-based and process-based arguments is demanded since "the former is concerned with getting the right product and the latter with getting the product right".

2.5 Reuse of Safety Artefacts

The main reuse approaches are available from the software development and many efforts to reuse software have succeeded; there is an increasingly overwhelming number of success stories available in literature [64, 65]. Nevertheless, the promises of decreased cost and increased dependability, and thus decreased risks, are not always realized. The frightening news about recent disasters definitely caused by careless soft-ware reuse are still being warningly associated with and attributed to all software reuse (e.g. in the Ariane project). Failure of a reused software component caused the loss of a rocket costing around half a billion dollars [66]. These recent disasters as a consequence of bad reuse on the one side and success stories as a consequence of good reuse on the other side are the key factors in deciding whether or not to enhance and sustain continued provision of reuse from a lucrative business perspective further on for safety-related systems. To sum up, before reusing a software component, the context and domain it was built for should be carefully compared with the context and domain it is intended to be built in, including the hardware and physical and organizational aspects [67].

In this thesis the focus is on two systematic reuse approaches: Product line engineering and different kinds of design patterns.

Product Line Engineering. The core idea of the product line approach is to build multiple products from a single infrastructure in a way that is aligned to stated business goals. An often used definition from Northrop and Clements [68] describes a software product line as "a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way." In distinction to other reuse approaches, software assets themselves contain explicit variability. The main concept is the differentiation of domain and application engineering. In domain engineering, reusable assets are developed together with a description of the supported variability. Concrete products can be derived in application engineering using these reusable assets in a predefined way. The most important issue is the description of variability. This is usually done with variation points and provide several possible variants, which can be chosen for a concrete product. At the moment a specific variant is selected, the variation point is said to be bound [69].

Patterns are used to solve similar problems with a general and universal solution. A well-known and proven solution for a specific problem is generalized so that it can be reused for similar recurring problems in other projects. Alexander describes the concept of using architecture patterns to solve similar problems in different projects [70]. The concept of patterns is used in many different domains including hardware and software. A good and very well-known reference is the book by Gamma et al. [71] (also known as the Gang of Four) had a significant impact on making the pattern approach popular for software development. The book includes general background and concepts as well as a collection of concrete patterns for object-oriented software design. The state-of-the-art provides a few dozen safety architecture patterns [72, 73], with some being just a variation of simpler ones. Armoush introduced in his PhD thesis [72] new safety patterns and provides a collection of existing safety patterns and a characterization of the main pattern representation attributes for embedded systems patterns (e.g. Name, Type, ID, Abstract, Context, Problem, Structure). These patterns are mostly based on the work of Douglas [74, 75] for hardware patterns and on Pullum [76] for software fault tolerance techniques brought into pattern notation for software patterns. Safety patterns usually include some kind of hardware redundancy, multiple channels with voters, or sanity checks [73].

Reuse and ISO 26262. Concerning reuse of ISO 26262 includes two means: One of them is "proven in use argument"[6] which is an alternate means of compliance with ISO 26262 that may be used in the case of reuse of existing items or elements when sufficient field data is available. The other one is "Safety Element out of Context (SEooC)"[7] , a safety-related element which is not developed for a specific item (e.g. not developed in the context of a particular vehicle). A SEooC can be a system/subsystem, a software component or a hardware component. An example of SEooC for software is an AUTOSAR software component.

Explorative study. As part of this thesis an explorative study was conducted in order to identify the state of practice of reuse in the context of different functional safety standards and development domains [77]. The study consists of a set of questions, which have been discussed with interviewees from companies of various domains. The companies

[6] See ISO 26262-Part 8, Chapter 14

[7] See ISO 26262-Part 10, Chapter 9

act in safety-critical domains with diverse product portfolios. Several points of view were covered by interviewing persons with different background and roles in the development process. The identified challenges are not just of technical character, but are also organizational and related to the used development process. In fact the used product line concept was often described to be missing a clear process, good documentation, clear variant management and good change management. Traceability through all the safety-related artifacts is a vital issue to manage all changes and analyze the impact of these changes. Small changes in the system specification require a complete and expensive re-work of the safety artifacts. Organizational challenges like management support for product lines and functional safety or establishing a common understanding of used terminology require long term strategies. The company specific interpretation of functional safety standards should be done by experienced engineers that provide clear and unambiguous guidelines and workflows for the development team of safety-critical products. So it should be clear what and why they have to do it and how it can be done in an efficient way.

Best practices for the development of safety-critical systems are typically starting with model-based development to support their development work on the one hand and on the other hand to enable a safety analysis and safety assessment. In many cases it was shown that experts are working on aligning the development process with the relevant functional safety standard. It has been shown, that people, which are involved in different projects, have different interpretations of the functional safety standards. A common interpretation is therefore necessary. The establishment of a safety culture in a company is taking longer time and is affecting not only E/E development but also management, verification, production and even maintenance. For example the experience of the product line concept described in literature is not applied within the industrial context for developing safety-critical products. Product lines have been developed over time within companies and different solutions have been emerged.

2.6 Potential for Improvements

In this section the potential for improvement are identified based on the related work summarized in the Sections 2.1 to 2.5 and a summary is provided in the following.

2.6.1 Safety Process Modelling (Improvement 1)

The related work concerning relevance of automotive standards and process modelling is presented in Section 2.1. The identified works in Section 2.2.1 describe a possible approach to combine quality and safety standards from a conceptual view. Furthermore, it shows process modelling by using a process modelling language for safety standard modelling. However an approach for combining quality and safety aspects in a systematic way for reusing process elements is not available to derive a process applicable in automotive development projects.

2.6.2 Safety System Modelling (Improvement 2)

Section 2.2 reviews relevant system modelling languages and discusses relevant safety analysis methods. SysML is identified as a standardized modelling language, that covers the demands for system modelling of different disciplines (e.g. software, hardware, mechanics) and can be extended to support the safety analysis demands. The Section 2.2.2 discusses existing approaches for possibilities of system modelling that supports different safety analysis methods. Nevertheless an establishment of a Model-Based Development Environment that uses SysML-based modelling and supports the specific safety analysis. In particular, safety analysis methods HAZOP, HARA and FMEA are demanded as a methodological and tool supported solution to support functional safety activities.

2.6.3 Early Safety Validation by Simulation (Improvement 3)

Section 2.3 discusses the possibility of simulation for early system safety validation. The related work regarding SystemC as a potential candidate for Simulation of safety aspects is discussed and the investigation of a FMI standard for Co-Simulation. However, the use of Co-Simulation with SystemC for the investigation of an workflow for FMI is identified as potential improvement by a combined SysML and SystemC approach for early safety validation.

2.6.4 Automotive Safety Argumentation (Improvement 4)

Section 2.4 reviews works for a safety case in different domains. In automotive domain the topic of safety case has been mentioned in ISO 26262 but there are no further requirements how to perform safety argumentation in a systematic way. A tool supported application of GSN approach for modelling of safety argumentation is demanded, where process- and product-specific aspects are taken into account.

2.6.5 Reuse of Safety Artifacts (Improvement 5)

The related work in Section 2.5 discusses examples and known issues, where reuse is performed and existing well-known reuse approaches are described. Furthermore, an interview survey was done, to identify the main challenges regarding reuse of safety artefacts from a practitioner point of view, which is in contradiction to the research perspective. Further investigations how to use such reuse approaches based on product-line engineering and reusable patterns for elaboration of safety artefacts are needed and are covered in that thesis.

2.7 Contributions to significance and added value

The goal of this thesis is to develop a supporting framework for the development of safety-critical automotive systems. The presented methodologies of this thesis takes (presented in Section 3) into account the mentioned potential improvements described in Section 2.6. The following main contributions are identified:

2.7.1 Process modelling framework (Contribution 1)

Design of a process modelling framework that covers relevant quality and safety standards and supports argumentation for process compliance covers improvement 1. Contribution 1 is presented in more detail in Section 3.1.

2.7.2 Support of Model-Based Safety Engineering (Contribution 2)

The Contribution 2 covers the Improvement 2, which develops a semi-formal system modelling approach based on SysML that provides all relevant safety artefacts to support relevant safety analysis methods, such as HAZOP, HARA and FMEA by specific spreadsheets or tools. Furthermore, Contribution 2 covers Potential Improvement 3 by providing co-simulation support for safety validation in the early design phases of the safety life cycle. Contribution 2 is describe in more detail in Section 3.2.

2.7.3 Support of Model-Based Safety Argumentation (Contribution 3)

Support of Model-Based Safety Argumentation covers Potential Improvement 4 Investigate possibilities for reuse approaches different kind of safety artefacts. Further on, Contribution 3 is exhaustively described in Section 3.3.

2.7.4 Investigation on approaches for reuse of safety artefacts (Contribution 4)

This Contribution 4 covers mainly potential improvement 5. In particular the reuse approach process-line engineering is investigated for quality and safety related process development in more detail in Section 3.1. Furthermore, different kind of pattern are elaborated and used in all methodological contributions of the thesis, nd applied in the methodologies with respect to Process Patterns in Section 3.1, Requirements Patterns and Architecture Patterns are describe in Section 3.2, and Argumentation Pattern are investigated in Section 3.3.

Chapter 3

Model-Based System Safety Engineering for Automotive Systems

The previous sections provide an introduction, explained the related work and highlighted the main challenges of the thesis regarding the development of safety-critical systems. This chapter describes the elaborated methodologies to overcome that challenges are described in more extend. Methodology 1 covers all aspects regarding Process Modelling and Management (see Section 3.1), Methodology 2 covers Model-Based Systems Engineering (MBSE) for Safety Artefacts, Safety Analysis and Simulation (see Section 3.2), and Methodology 3 shows the support for Safety Argumentation and Safety/Security Co-Engineering (see Section 3.3).

3.1 Methodology 1: Process Modelling and Management

3.1.1 Process Modelling Phases

The main intention of process modelling is the ability of reuse of relevant process elements. For that reason a process line approach is beneficial since commonalities and variabilities can be clearly systematized. As mentioned in the related work section, currently, there is no satisfying modelling language nor a tool is available supporting the process lines. However, the modelling language SPEM2.0 and the tool EPF Composer have been identified to be sufficient to implement the methodological framework based on safety-oriented process line and show its validity. In the following the approach is described with help of the exemplary application of process aspects regarding quality (ASPICE) and safety standards (IEC 61508 and ISO 26262). The commonalities and variabilities can be modelled in two phases which are *Phase A: Domain Engineering* and *Phase B: Process Engineering* [78]. Domain Engineering exploits standard compliant process, identifies commonalities and creates base processes . Process Engineering exploits commonalities and variabilities to derive standard-specific single processes.

An overview of the overall process modelling and management workflow is shown in Figure 3.1. Seven steps have been defined in Methodology 1 (M1) to derive the process modelling approach.

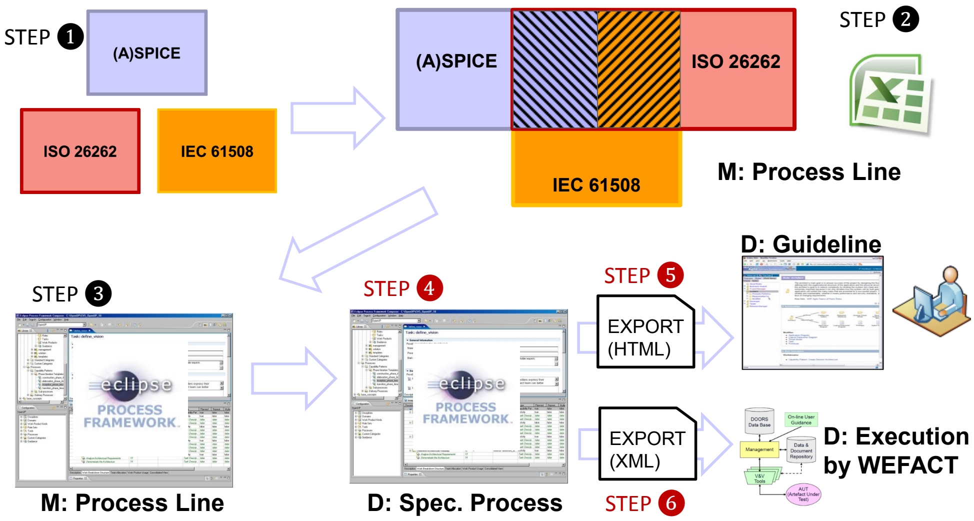


Figure 3.1: Overview of Process Modelling by using SoPL

M1- Step 1: Determine different standards with commonality. In this step a systematic analysis of the specific standards, e.g. ISO 26262, have to be done. Hence, the ISO 26262 standard defines more than 100 work products and more than 1300 recommendations on these work products. The challenge is to perform project tailoring according to the ASIL, identify the requirements to be fulfilled and the methods to be used for the elaboration of each work product. For the approach a spreadsheet lists all 1300 recommendations demanded by ISO 26262 and links them to relevant work products. Since recommendations and work products are organized in a matrix, each recommendation can be assigned to one or more work products by filling out the crossing cell with an attribute of the specified dependency. Filtering capabilities enable to focus on one dedicated work product for one specific ASIL, thus efficiently identifying the work to be performed for any specific work product. The same approach has to be done for the standards IEC 61508 and ASPICE. During this step the definition of a common terminology has to be performed, a mapping of the activity naming and definition of the identifier of a particular activity has to be done.

M1- Step 2: Identify commonality of process elements. Processes are constituted of phases, which in turns are constituted of a set of activities. Activities consists of a set of tasks. Finally, tasks are constituted of a set of steps. Thus, commonalities are unlikely at the root level of this nested structure. From an execution point of view, phases, activities, tasks, etc. may be performed in a different order. From a pure syntactical comparison, all these differences may be interpreted as variabilities. However, to be able to justify a process line approach the amount of commonalities must be greater than the amount of variabilities. To reduce variabilities and increase commonalities, the following definitions

are at disposal:
The ***Commonality Types*** are differentiated by *partial* and *full commonality.*
Partial commonality: whenever process elements of the same type expose at least one common aspect (e.g. at least a step is equivalent). In this approach, this definition is used in a flexible way. When comparing process elements of the same type, either the entire set of processes (process line) is considered or subsets of them. More specifically, the heterogeneous set of standards examined is divided into two subsets: one containing the non-safety-related standard (ASPICE) and the second containing the safety-related standards (ISO 26262 and IEC 61508). This flexible usage provides the potential to create a greater extent of reusable process elements.
Full commonality: whenever process elements of the same type expose only common aspects (e.g. all steps are equivalent). For the sake of terminological completeness, we also clarify that a process variant is a representation of a particular instance of a variable process element of the real world or a variable property of such an element.

Variant elements. Base elements also include reusable standard-specific variants. These elements are named as variants and they are obtained by enriching the elements representing partial commonalities. For the complete creation of a process line for each process element (e.g. task, work product, guideline, etc.), several variants (e.g. standard-specific, company-specific, project-specific, etc.) should be provided. Typical standard-specific variants are those that deal with different safety integrity levels (SIL or ASIL). Thus, the variants also includes process elements that are not predetermined by a standard. For the creation of company-specific as well as project-specific process elements, standard-specific partial commonalities should be enriched or replaced.

Optional elements. Elements that might be standard-specific and that do not represent a mandatory element for each process of the process line. Optional elements can be replaced by an empty element if the single process to be derived from the process line does not include it.

M1- Step 3: Development of reusable process elements in a repository. Based on the elaborated spreadsheet a standard compliant process model is created with EPF-C. The process model creation by EPF-C is described with further illustrations in [78].

M1- Step 4: Company specific and project specific extension of process elements. In this step any required adaptations of company specific elements based on the standard compliant base models are created. For example specific entities, which are not demanded by standards are added, such as roles, tools and guidance elements. Furthermore, project specific representations of the model can be derived and further specific elements are added. Company specific methods can be selected from a provided set and project specific work products can be defined.

M1- Step 5: Creation of a process guideline. The tool EPF-C provides the possibility to export specific process models in a so called delivery process. It is possible to choose specific process elements in the project browser, which are relevant for the specific needs of the user. The exported process model can be used as guidance for involved engineers in HTML standard format, which can be used by any conventional browser.

M1- Step 6: Execution of the modelled process. EPF-C model supports to export the process model in XML format. A specific mapping of all process elements from EPF-C to WEFACT was done. Process Execution and process management is performed by WEFACT, where defined roles, methods and tools are assigned to activities. Specific activities can be executed in WEFACT and the status of created work products can be followed and artefact can managed in a user defined repository.

3.1.2 Implementation of Process Modelling

This section describes the used tools for implementing the process modelling methodology.

EPF-Composer (EPF-C). The open source tool EPF-Composer[1] implements modelling of the SPEM 2.0 process modelling language. The main goals of the Eclipse Process Framework Project are (1) to provide an extensible framework and exemplary tools for software process engineering - method and process authoring, library management, configuring and publishing a process, and (2) to provide exemplary and extensible process content for a range of software development and management processes supporting iterative, agile, and incremental development, and applicable to a broad set of development platforms and applications [2].

Workflow Engine For Analysis, Certification and Test (WEFACT). The tool WEFACT [79] originated from the DECOS Test Bench [80], which was a web-based distributed platform for requirements-based testing with continuous impact assessment in order to support the safety case with evidences. In SafeCer, the test workflow was extended to a workflow for safety certification and in EMC2 the quality attribute of security was integrated. The basis for defining the WEFACT workflow is a process model, which is created using the EPF-C. With this tool, the previously modelled generic process flow, which reflects the safety and security requirements of the applicable standards, is tailored to the domain- and customer-specific practices as well as to the needs of the individual project. The resulting specific process model is then imported from EPF-C into WEFACT, and for each requirement so-called "V&V Activities", typically test or analysis activities, are defined, which apply a V&V tool to an AUT (Artefact Under Test). The resulting evidences for respective requirements contribute to the safety and security case via a report generation tool. All artefacts are linked to each other by full traceability management enabling WEFACT to perform workflow management through continuous impact management.

[1] Eclipse Process Framework, www.eclipse.org/epf/

[2] https://en.wikipedia.org/wiki/Eclipse_Process_Framework

3.2 Methodology 2: MBSE for Safety Artefacts, Safety Analysis and Simulation

During the design stage of development, the system model provides an appropriate architectural design, which usually implements functional and technical requirements. The system design is about the breakdown of large systems into smaller subsystems to reduce the existing system complexity. The elaborated system modelling elements and structures provide the basis for safety analysis activities (see Section 3.2.1 and Section 3.2.2). Furthermore, for code generation templates related to blocks, which translate architectural decisions into simulation models for early design validation (see Section 3.2.3 and Section 3.2.4).

3.2.1 Safety Artefact Modelling and Analysis

The system model covers all kinds of interactions and influences between system and vehicle behavior. The safety-related artefacts of the system are modelled by using the semi-formal notation SysML. To this end, the SysML model shows the connections between components, functions, malfunctions on different system levels. Each component is represented by a functional block. Functions and malfunctions are represented by use case elements. Concerning structure, the SysML model supports the definition of block definition diagrams and internal block diagrams. Especially internal block diagrams contain information how blocks are connected and which interfaces they share. One aim of this system model is to create the three main nets for safety analysis: the structure net, the function net and the failure net.

Following steps are performed during Methodology 2 (M2), MBSE approach for artefact modelling and analysis in the concept phase:

- **M2- Step 1**: Safety Modelling by Enterprise Architect (Functional Architecture)
- **M2- Step 2**: Safety Analysis by HAZOP
  - Step 2.1: Data-Exchange EA→HAZOP (Export Structure and Functions)
  - Step 2.2: Safety Analysis by HAZOP (Perform HAZOP by spreadsheet)
  - Step 2.3: Data-Exchange EA←HAZOP (Add Fault - Error - Failure - Hazard)
- **M2- Step 3**: Safety Analysis by HARA spreadsheet
  - Step 3.1: Data-Exchange EA→HARA (Export Hazards and Failures)
  - Step 3.2: Safety Analysis by HARA (Perform HARA by spreadsheet)
  - Step 3.3: Data-Exchange EA←HARA (Add Safety Goals)
- **M2- Step 4**: Safety Analysis by FMEA tool APIS
  - Step 4.1: Data-Exchange EA→FMEA (Export Structure-, Function-, Failure-Net)
  - Step 4.2: Safety Analysis by FMEA (Perform FMEA by APIS IQ FMEA)
  - Step 4.3: Data-Exchange EA←FMEA (Add Preventive and Detection Action)

- **M2- Step 5**: Derive Functional Safety Requirements from Preventive and Detection Action

Figure 3.2 provides a brief overview of the interaction points of the system model and the used safety analysis methods. The numbers in the blocks show the order of performance of safety analysis activities.

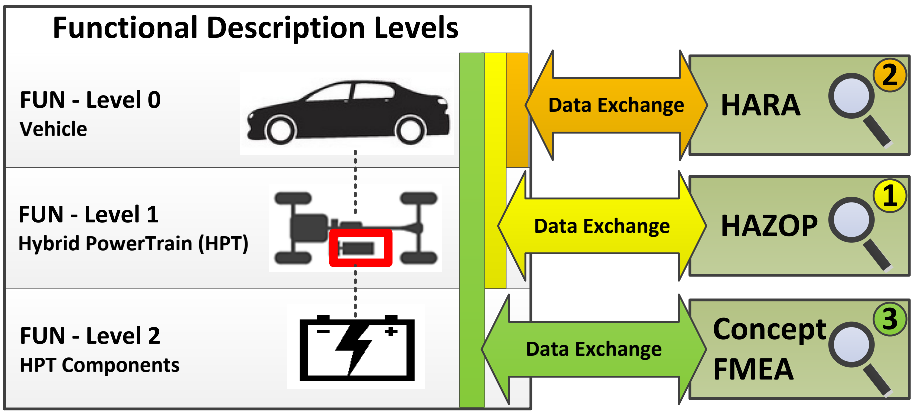


Figure 3.2: Dataexchange of Safety Model and Analysis Methods

The system modelling and analysis methodology is described in more detail in the SAE conference paper [81].

3.2.2 Implementation of Safety Artefact Modelling

Enterprise Architect for Safety Modelling. For the implementation of the MBSE methodology the tool Enterprise Architect (EA) provided by SparxSystems[3] was used for the safety modelling based on SysML. EA is a commercial modelling tool that provides multi-user support and it is graphical tool designed to build robust and maintainable software. The tool features flexible and high quality documentation output and provides possibilities for specific tool extensions by so-called Model-Driven Generation (MDG) technologies. These MDG technologies allow users to extend EA's modelling capabilities to specific domains and notations.

HAZOP Spreadsheet. The HAZOP analysis was implemented by a customized HAZOP spreadsheet in MS-Excel, which supports the main activities of the HAZOP analysis method described in Section 2.2:

- Use structural element (component) and allocated functions as basis for the HAZOP
- Use of guidewords to derive malfunctions on component level
- Derive malfunction behaviour and hazards on vehicle level

[3] https://www.sparxsystems.eu

HARA Spreadsheet. The ISO 26262 demands the performance of the HARA. For the performance of that HARA approach a specific spreadsheet in MS-Excel was developed, which provides following features:

- Situation analysis supports identification of specific vehicle Driving Situations
- Import malfunction behaviour and hazards on vehicle level as basis for HARA
- Derive hazardous events by combination of driving situations and hazards
- Perform prioritization of hazardous events by filtering
- Each hazardous event is classified by three parameters Severity (S), Exposure (E) and Controllability (C)
- Determination of ASIL based on risk graph of ISO 26262
- Elaboration of safety goals in correspondence with hazardous events

FMEA by tool provided by APIS. APIS IQ FMEA PRO [4] is an commercial, industrial established and TÜV certified tool for functional safety analysis for different types of Failure Modes and Effects Analysis (FMEA) such as Concept-/Design-/Process-FMEA, FMEDA, and Fault Tree Analysis (FTA). The APIS tool is used for performing the Concept-FMEA in the Concept Phase and the Design-FMEA on System Level.

Safety-Data-Exchange. For the data exchange between SysML model and safety analysis activity a specific prototype tool called "Safety-Data-Exchange" has been developed and implemented within the safety team of VIRTUAL VEHICLE[5]. The tool is programmed in java in an Eclipse software development environment and provides following features:

- Bidirectional Data-Exchange between EA and HAZOP spreadsheet:
  - Export of structural element (component) and functions from system model (EA) to HAZOP spreadsheet
  - Export of malfunctions (incl. guidewords), malfunction behaviour, hazards from HAZOP and import to EA to extend existing system modelling elements
- Bidirectional data-exchange between system model in EA and HARA spreadsheet:
  - Export of hazards and malfunction behaviour at vehicle level from system model (EA) to HARA spreadsheet
  - Export of safety goals with associated ASIL incl. safe state from HARA and import to EA to extend existing system model
- Bidirectional data-exchange between system model in EA and FMEA tool:

[4] Tool vendor APIS Informationstechnologien GmbH https://www.apis-iq.com

[5] Kompetenzzentrum – Das virtuelle Fahrzeug Forschungsgesellschaft mbH

- Select component for FMEA: Specific component within the vehicle structure can be chosen to create an FMEA formsheet
- Export required data of the system model from EA
- Import three data nets of structure, function and malfunction as a basis to perform the FMEA
- Export preventive and detection action from FMEA
- Import preventive and detection action and associate to the malfunction in the system model in EA

3.2.3 MBSE Simulation Support

This section discusses how a combined SysML and SystemC approach. The system model in SysML is used for code generation templates related to blocks, which translate architectural decisions into simulation by using SystemC modules. These models are connected with unique signals and channels. The logical link between SysML model blocks and SystemC modules is formalized and can be traced throughout the entire design process.

Co-Simulation by SystemC. As depicted in Figure 3.3, has the potential to support the needs for efficient design and validation of safety relevant embedded systems.

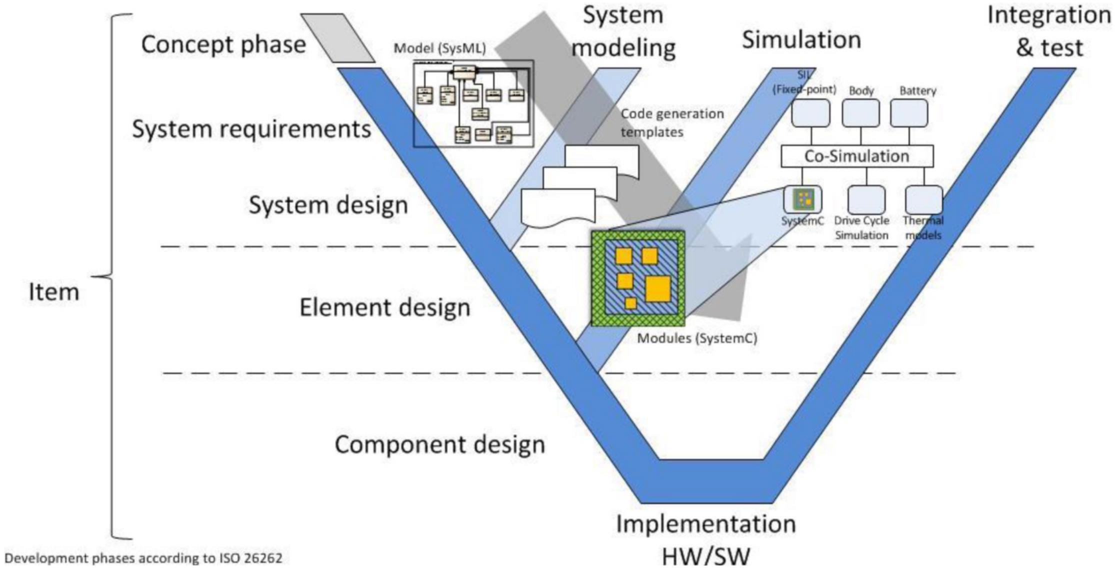


Figure 3.3: Development Process Overview

This four-staged meta-model can be applied to generate arbitrary composites of embedded systems. However, what ISO 26262 refers to the term architecture is usually defined by three of them, namely item, system and element. The model supports this structuring by extending block definitions. This way, entity boundaries can be drawn arbitrary between different engineering domains. The rightmost hand side of the V-model covers tasks concerning integration and test. While integration of software components or hardware units can be done on supplier level, integration tasks on vehicle level require the availability of a prototype car. For several reasons we do not strive for vehicle level integration - besides availability, prototypes have shown to be costly. Every stage of development, which can be accomplished using simulation, helps to save time and cost. Thus, the rightmost integration and test phases in the V-model are spared and a parallel arm covering simulation techniques is created. Since the focus is on system modeling and design, another arm covering these topics is drawn in parallel.

Standard compliant Co-Simulation Models. This section presents a tool-independent method on how to integrate electric and electronic system models together with their corresponding simulation engines into single Functional Mock-up Units (FMUs) implementing the FMI. Aforementioned models are built using SystemC and SystemC-AMS. By doing so, SystemC becomes available to a broad range of applications on system level in a standardized manner. The resulting FMUs are highly transportable and may easily be integrated within larger and more complex co-simulation scenarios for fast and convenient information exchange and system verification.

In order to integrate and execute SystemC and SystemC-AMS simulation models in context of an FMU, a structured method is proposed. The necessary steps are illustrated in Figure 3.4, indicated by the dashed box and described as follows:

(A) Modelling and simulation of a single component model.

(B) Model interface identification for coupling to the co-simulation environment.

(C) Wrapper class specification for controlling the model interface.

(D) C-interface specification for FMI integration.

(E) FMI integration using a predefined software developer kit.

(F) FMU compilation and assembly together with (architectural) model description.

(G) Integration of FMU to co-simulation master for simulation based system level verification.

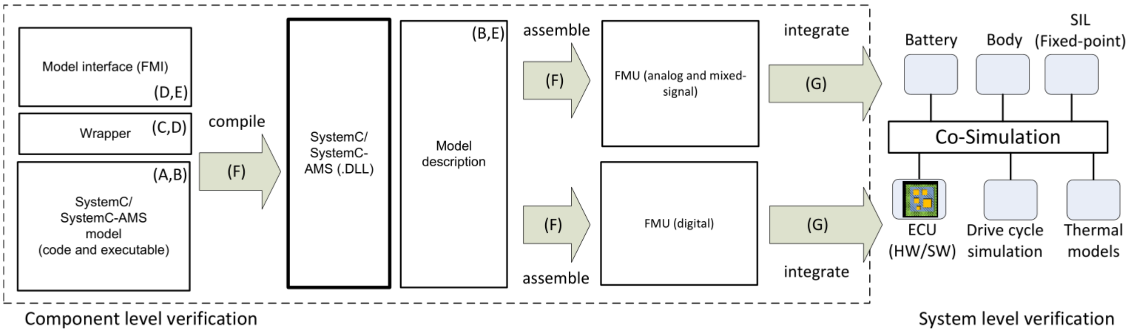


Figure 3.4: Proposed process for integration of executable simulation models into FMUs for co-simulation

Subsequently, each step is explained in detail in the referred publication [82].

3.2.4 Implementation Simulation Support

Simulation engine for SystemC. The previously explained common V-diagram as a process model does not make any assumptions on tooling, since SysML and SystemC [42] are pure languages. In this section, a workflow is proposed, mainly based on the strong connection between SysML, SystemC and co-simulation. Based on preliminary architectural assumptions, requirements and safety integrity levels are assigned to their corresponding architectural entities. To accomplish these steps, Enterprise Architect was chosen as primary modeling tool for SysML. The various types of requirements can be collected, structured and regrouped within a system model. Its export capability to XMI data format and the application of transformation templates allows the perpetuation of architectural decisions from SysML to SystemC language. By compiling the SystemC code, the Microsoft C++ compiler builds the simulation engine for SystemC. At this stage the preliminary architectural assumptions and system design results turn into concrete executable models for the first time in the proposed V-model. This point has a second advantage. By receiving an executable C++-model, an interface to co-simulation methodology is achieved at the same time.

Independent CO-Simulation (ICOS). The idea of independent co-simulation is supported, by introducing the in-house (of VIRTUAL VEHICLE[6]) developed co-simulation software ICOS [11]. This way, the inclusion of domain specific tools to the V-model, becomes possible, even on higher levels of abstraction. In the system model, the behavior of components and elements can only be expressed at a very high level of abstraction. The process transition to SystemC code at this stage enables two benefits related to co-simulation. First, it becomes possible to include other relevant models. These models can be existing concrete implementations, thus the idea of component re-use is supported. Second, it becomes possible to replace existing, abstract SystemC models with more complex models. A combination of both possibilities not only allows the evaluation of high-level concepts in a broader environment, but also the traceability of certain features, back through the chain. The fulfilment of safety goals can be evaluated this way.

3.3 Methodology 3: Support for Safety Argumentation and Safety/Security Co-Engineering

3.3.1 Safety Argumentation

The elaborated modelling support for safety argumentation provides following features:

- Semi-formal modelling of Safety Argumentation by Goal Structuring Notation
- Process- and Product-based Argumentation
- Reuse by using argumentation pattern

Each of these features are described in the following in a certain extend.

[6] Kompetenzzentrum – Das virtuelle Fahrzeug Forschungsgesellschaft mbH

Semi-formal modelling of Safety Argumentation by Goal Structuring Notation. In Goal Structuring Notation (GSN), an argument is defined as a series of connected claims. The main elements of the GSN are goal, strategy, solution, context and module illustrated in Figure 3.5. Strategy-elements are used to declare reasoning behind the connection between goals and sub-goals. Context-elements provide additional information to support a correct understanding of a specific argumentation part. Solutions are elements that support goals because they document pieces of evidence. The relationship between GSN elements is documented in a graphical way using different linkage elements (arrows). The two types of linkage elements are 'SupportedBy' and 'InContextOf'. The former, represented by lines with solid arrowheads, indicates inferential or evidential relationship, the later represented as lines with hollow arrowheads, declares contextual relationships. Modules are used to hide detailed structures and simplify goal structures to provide a general view. Between modules both types of linkage are possible.

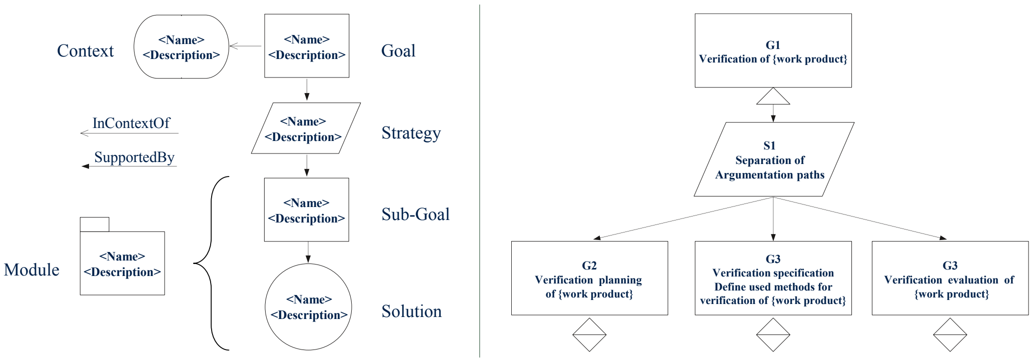


Figure 3.5: Basic argumentation elements based on GSN standard [83]

Process- and Product-based Argument. To deliver proof of functional safety for a defined development phase all requirements demanded by a standard (e.g. ISO 26262) have to be covered. Two different types of argumentation have to be considered, namely process-based and product-based argumentation. The proposed methodology defines each type of argumentation separately although they stay in direct relationship in the line of argument. Product development forces an established engineering process, supported by joint argumentation.

In case of process-based argumentation the arguments are directly associated with company specific processes which are derived from the ASPICE process reference model as well as the ISO 26262 safety life cycle. Process-based argumentation provides arguments to prove that the defined process fulfils demanded requirements. The argumentation is based on the existence of needed work products but not on their content. The approach in case of process-based argumentation is to document arguments, which support the process, in parallel with the process development. The process argumentation contains reasons why a particular process task has to be done in the described way. GSN elements like strategy and context are used to explain the decision why a goal splitting was done. Information about decisions is needed for process audits therefore it should always be documented.

The GSN notation uses the possibility of unrestricted formulation to discriminate from a generic process and to emphasize arguments why deviation is needed.

The other argumentation type is product-based. That kind of argumentation is elaborated based on content of available work products. With help of these work products it must be possible to establish an argument that the developed product is safe in terms of the relevant standards. Before project release for production a functional safety assessment has to be passed and arguments have to be available in a way that an external assessor can comprehend them. The focus of attention is to provide arguments why particular product-related, technical decisions have been made and why specific methods or tools have been used. Within a generic formulated process a product specific decision determines a branch-off point. A decision based on a product specific requirement causes the necessity for different safety measures, which are related to different software and hardware to manage a specific system. At that time the process becomes product-requirement-driven.

Reuse by using Argumentation Pattern. A pattern provides templates, guidance and formalisms to create goal structures for previously defined processes or products. The definitions concerning patterns and templates are based on [16] and additional structural details of patterns which are defined in [6]. The most relevant attributes of patterns (based on [6]) are Intent of the pattern, Template, Motivation, Applicability, Pitfalls Consequences. The objective of patterns is to support standard compliant safety argumentation and best practices from previous projects. Additionally it should be designed to be extendable and adaptable based on lessons learned. Patterns assist users by providing predefined elements, which are adaptable for tailoring needs. The pattern is not present in the final argumentation.

A workflow for reusing of argumentation patterns is introduced that covers three subsequent phases (see Figure 3.6)

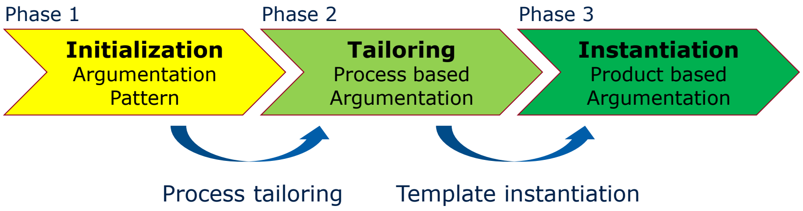


Figure 3.6: Phases to create reuseable process- and product-based arguments [83]

Phase 1 - Initialization of Development Process. The initialization phase is used to prepare all needed process elements to design a complete standard compliant development process. Activities in this phase are selection of relevant standards as well as identification of existing process and argumentation patterns which are suitable for reuse. The company specific process and the accompanying argumentation pattern are outcomes of this phase.

Phase 2 - Tailoring for process-based Argumentation. The tailoring from the company specific process to the project specific process means that process elements are selected to form the project specific development process. This selection includes the corresponding argumentation templates provided by patterns as well as methods and tools which should be used in the project. Creating a project specific process deals with decisions and judgements dependent on ASIL and needs expert knowledge. Process-based argumentation is needed for functional safety audits.

Phase 3 - Instantiation for product-based Argumentation. This phase covers product development by executing the project specific process. Templates for product-based argumentation support product specific decisions for a defined product. These decisions are made once and they are put into practice for a complete product line of a system. The generic template provides argumentation which is typically valid for systems (e.g. specific physical parameters like voltage or temperature). The complete argumentation structure is achieved by instantiation of the template to product specific context. The demand of a complete safety case is the main reason to elaborate product-based arguments for functional safety assessment. With help of results documented in work products it becomes easy to argue that product specific claims are valid. This argumentation is done bottom up starting with results of the development process. Furthermore, it is important to have quick access to related evidence that proves a product is safe.

The complete description of the safety argumentation approach is given in the conference paper [83].

3.3.2 Implementation of Safety Argumentation

OpenCert [7] is an open source tool for product and process assurance/certification management to support the compliance assessment and certification of safety-critical systems in sectors such as aerospace, railway and automotive. OpenCert was originally created as a result of the FP7 project OPENCOSS [8]. The main tool functionalities include:

- Knowledge Management from Standards - This feature deals with knowledge management, captures information from standards, e.g. interpretations about intents.
- Assurance Project Management - It factorizes aspects such as the creation of assurance projects. This module manages a “project repository”, which can be accessed by the other modules
- Argumentation Management - This feature manages argumentation information applying GSN graphical notation. It also includes mechanisms to support compositional safety assurance, and assurance patterns management.
- Evidence Management - It manages the full life-cycle of evidences and evidence chains. In addition, this module is in charge of communicating with external engineering tools (requirements management, implementation, V&V, etc.)

This OpenCert toolset has already been used in avionics [84] and automotive domains [85], where it was called PROSSURANCE.

[7] https://www.polarsys.org/proposals/opencert

[8] OPENCOSS, http://www.opencoss-project.eu/node/7

3.3.3 Safety and Security Co-Engineering Framework

Extends previous described methodologies regarding process modelling and argumentation regarding security aspects.

The elaborated Co-Engineering Framework provides following features:

- Systematic reuse of process elements
- Process execution based on process model
- Safety argumentation for process-related aspects
- Analysis method for joint safety and security consideration

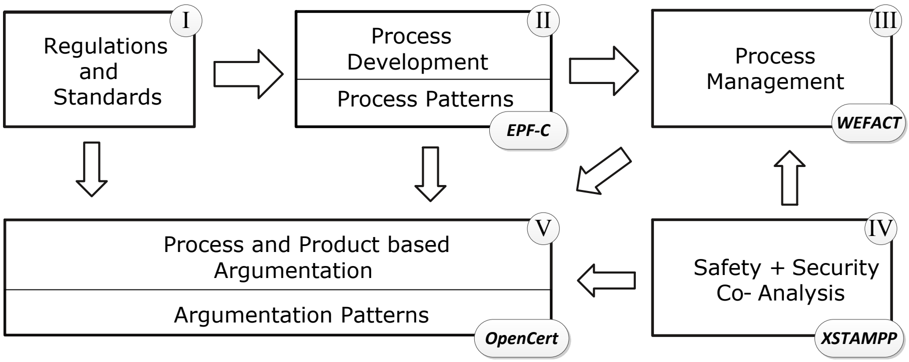


Figure 3.7: Process Modelling Framework

The main steps of the proposed methodology, which considers all process steps necessary in an automotive safety and security related development project, are shown in Figure 3.7:

Regulations and Standards (I). In a first step all relevant regulations and standards are identified. The automotive use case deals with ISO 26262 regarding road vehicles functional safety and SAE J3061.

Process Definition (II). It is challenging to match these two topics because they are influencing each other. Process definition has to consider that elaborated process steps are not only in parallel but also highly interactive, especially when functional safety and cybersecurity have to be handled. In addition, processes have to incorporate special analysis methods, which handle safety and security aspects in one common analysis methodology. Integrated processes, which are basis for co-engineering, unite safety with security activities. They lead to integrated requirements, work products and argumentation.

Process Management (III). The core of the framework is the distinction between functional safety and security related process and product requirements and the identification of interactions. Process requirements describe activities and steps, which are demanded by standards, while product requirements are derived from the system under development. In order to manage the processes and support the process execution, appropriate tools are useful, which assist developers with requirement and work product management. Work products are process outcomes representing different types of evidence. Evidence shows capability and maturity of the development process, compliance to the underlying standards and safety as well as security of the developed products. In addition, evidence is used to support arguments which are related to requirements.

Safety and Security Co-Analysis (IV). The intention of the proposed framework is to integrate functional safety and security. For that reason special analysis methods, which handle safety and security aspects in one common analysis (co-analysis) methodology, have to be used.

Process- and Product-based Argumentation (V). Consequently the argumentation demonstrates that the item under consideration contains no unreasonable risk and consolidates functional safety and security. To visualize these relationships between requirements and work products GSN is used. A more detailed description of the argumentation approach can be found in [10, 83].
To recapitulate a loop (depicted in Figure 3.7) in which every activity is supported by a tool is considered: The created processes are modelled, instantiated and executed. The process output is evidence to argue that activities for the development of a specific product have been performed and are compliant to specific regulations. Once the process has integrated various disciplines, like safety and security, project managers have support to coordinate their cooperative actions.

The complete description of the Safety and Security Co-Engineering Framework methodology can be found in the conference paper [86].

3.3.4 Implementation of Co-Engineering Framework

EPF-C is used for tool-support regarding the safety and security process modelling. It supports the SoPL approach and provides export function of the process model in XMI format (see also Section 3.1.2).

WEFACT , web-based distributed platform for requirements-based testing with continuous impact assessment in order to support the safety case with evidences. WEFACT provides import function for process models in XMI format from EPF-C. Test workflow is extended to a workflow for safety certification and the attribute of security is integrated. Furthermore, simulation tools can be integrated in WEFACT, executed by an user front-end and the simulation results are stored in a central repository (see also Section 3.1.2).

OpenCert is an open source tool for product and process assurance/certification management to support the compliance assessment and certification of safety-critical systems in sectors such as aerospace, railway and automotive. OpenCert supports creation of GSN structures and mapping of evidence to requirements demanded by underlying standards (see also Section 3.3.2).

XSTAMPP (eXtensible STAMP Platform) [87]. STPA (Systems-Theoretic Processes Analysis) is a new hazard analysis technique based on STAMP. STPA is already being used in different industrial domains (e.g. space, aviation, medical or automotive)The extensible STAMP platform called XSTAMPP as tool support designed specifically to serve the widespread adoption and use of STPA in different areas, to facilitate STPA application to different systems and to be easily extended to include different requirements and features. XSTAMPP is an Eclipse RCP 4 based tool which guides users through the Safety and Security Co-analysis by STPA-Sec process and supports the modelling of control loops and the definition of constraints.

In following Chapter 4 the evaluation of results gathered during application of the co-engineering approach for an automotive systems are described in more detail.

Chapter 4

Experimental Evaluation

This section shows the results of experimental evaluation of the presented methodologies through application of the framework to different development use case scenarios with respect to automotive high voltage battery system case study.

4.1 Description of High Voltage Battery Systems

High Voltage (HV) battery systems are a central part of battery-powered Hybrid Electric Vehicles (HEVs), Plug-In HEVs (PHEVs), or Electric Vehicles (EVs), which are becoming more and more important. One reason is the high energy efficiency of E/E systems and the zero (local) environmental pollution of EVs. The main disadvantage is the relatively short operation range, which is far less competitive compared to conventional vehicles with internal combustion engines. Conventional vehicles provide good performance and long operating ranges by utilizing the high energy-density advantages of petroleum fuels. HEVs combine the advantages of both technologies. Some of the main targets for batteries to be used in HEVs are low costs, high power density (e.g. 1,200W/kg), very high cycle life time (e.g. 200,000 cycles of charge/discharge), long life time (e.g. 9 years), and safety [88].

4.1.1 HV Battery System for HEV Powertrain

With the growing importance of e-mobility, automotive HV battery systems are becoming more important as well. High power (e.g HEV up to 250kW to provide more dynamic driving torques) and high energy application (e.g. EVs such as Tesla Model S up to 75kWh [1] to allow longer driving distances) are already being applied in series-production vehicles. Increasing power and energy while decreasing the battery geometries leads to an increase of potential critical effects in the case of malfunctions. In particular, e-mobility is highly interdisciplinary, whereby risk reduction also results from different technical disciplines (e.g. mechanics, chemistry). This means that system safety has to consist of different safety disciplines as well (i.e. functional, electrical, mechanical, and chemical safety). One example for electrical safety could be the prevention of hazardous voltage through the use of galvanic disconnections or isolation. Mechanical safety aims to prevent the deformation of the battery in the case of an accident through the use of cell housings

[1] https://www.tesla.com/models

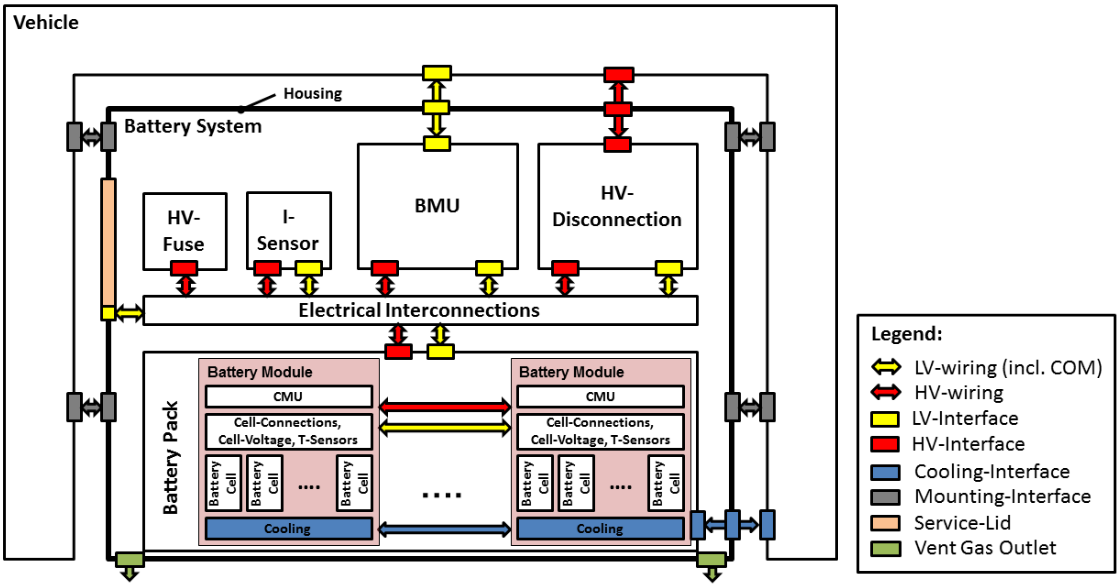


Figure 4.1: HV battery system architecture [88]

or the installation location for example. Chemical safety can prevent explosions or fire by using a mechanical venting outlet for toxic gases. Functional Safety is related to any safety critical malfunctioning behaviour of any system that may harm people. However, all of these measures are applicable for the development of a safe system, such as HV battery systems.

Figure 4.1 shows a schematic representation of a system architecture of an HV Lithium-Ion (LiIon) battery system. The BMS is the main E/E system inside of an HV battery to power EV or HEV. The BMS consists of several input sensors for cell voltages, cell temperatures, output current, output voltage, and actuators like HV contactors for disconnection. This system is connected to various powertrain control units, the charging interface (enabling the communication with battery charging stations), the on-board diagnostic interface, and via a dedicated gateway to the vehicle infotainment systems (including the human machine interface and a wireless infotainment internet connection).
The main functions of the HV battery system are:

- Provide electrical energy
- Store/charge electrical energy
- Electrical management of the battery system

Possible malfunctions are:

- Deep discharging of battery cells
- Overheating of battery cells
- Charging by deep discharged battery cells

- Overcharging of battery cells

These malfunctions could lead to the following possible hazards, that may harm people:

- Hazardous voltage
- Cell leakage/venting gas
- Fire/explosion

A more detailed description of the battery system and its components can be found in [88].

4.1.2 Overview of Use Case Scenarios

The following use case scenarios show the applicability of the elaborated methodologies in Chapter 3:

- Use Case Scenario 1: Executable Process Model that covers Quality and Safety
- Use Case Scenario 2: Modelling, Analysis and Simulation of Safety Artefacts
- Use Case Scenario 3: Application of Safety and Security Co-Engineering and Argumentation Framework

4.2 Use Case Scenario 1: Executable Process Model that covers Quality and Safety

This use case scenario covers following aspects:

- Demonstration of Methodology 1
- Cross domain scenario for industrial and automotive applications
- Relevant Standards: Quality - (A)SPICE, Safety - IEC 61508 and ISO 26262
- Re-use integrated process model for quality and safety standards (different domains)

4.2.1 Description Use Case Scenario 1

Industrial suppliers for different domains (=cross domains) are forced to apply different safety standards for the development of safety critical products. This use case scenario demonstrates the process modelling approach based on SoPL described in Chapter 3.1, which covers quality demanded by ASPICE and safety standards (ISO 26262 and IEC 61508).

As an example the system design phase is highlighted for the description: In ASPICE the majority of the activities concerning system design is part of the process ENG.3 System architectural design. Within this process the Process Purpose, the Process Outcomes and Output Work Products are defined. ISO 26262 is oriented at the ASPICE structure, where *part 4 - chapter 7* covers the system design (entitled System design). This chapter includes objectives, input from other activities, requirements for the activities and work products. The structure of IEC 61508 differs significantly from ASPICE. IEC 61508 does not describe what should be done but sets out objectives and requirements for the activities. The system design phase is covered mainly in *chapter 7.2 - E/E/PE system design requirements specification* and *chapter 7.4 - E/E/PE system design and development* of part two. Work products are not mentioned in the normative part.

In the following the two phases for Domain and Process Engineering are described.

Phase A: Domain Engineering As an example of elements of type *partial commonality*, the task "Define system architectural design" is considered. This task is present in all three standards. In ISO 26262 it is called "System design specification", in ASPICE it is called "Define system architectural design" and in IEC 61508 it is called "E/E/PE system design and development".

Phase B: Process Engineering To create a delivery process concerning ASPICE and ISO 26262 (ASPICE_ISO26262_System_Design_Delivery_Process), it is necessary to add all the needed process elements. This means that all elements in the EPF-C model in the method content *Full* and *Partial* must be added to the *Work Breakdown Structure.* Which elements from the method content *Optional* are added depends on the considered standards. If the focus is on the combination of ASPICE and ISO 26262 the elements concerning these two standards have to be added. At this point a delivery process is established which is compliant with two desired standards. Figure 4.2 shows an illustration of the results for ASPICE and ISO 26262.

More details about the application of the methodology in the specific use case scenario can be found in [78].

4.2.2 Results of Use Case Scenario 1

Reusable Process Model. The modelling of process elements for Quality and Safety in SPEM2.0 by the tool EPF-C can be reused for specific projects in industry (IEC 61508) and automotive (ISO 26262) domain.

Engineering Guidance. The elaborated process model can be exported in a general web-browser compatible HTML format to provide engineering guidance and support specific roles in a project.

Process Execution. The process model can be exported in XML format and for process execution in the tool WEFACT a standard compliant, reference process model for project execution and management.

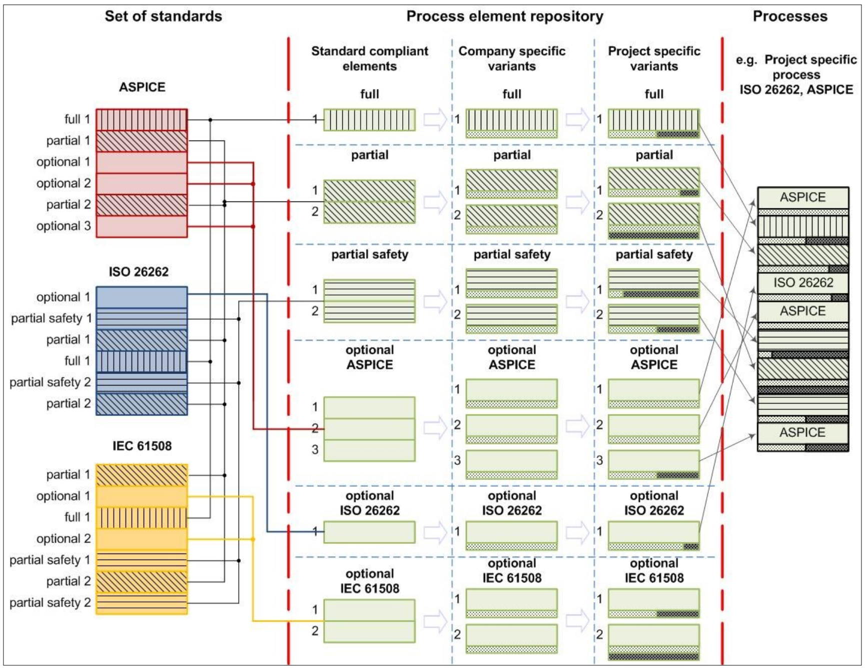


Figure 4.2: Demonstration of SoPL approach in use case scenario 1

4.2.3 Lessons Learnt of Use Case Scenario 1

General soundness. The elaborated adoption of the SoPL approach is applicable since prescriptive processes mandated by the standards exhibit commonalities. The adoption is also beneficial since it enables systematic reuse of process elements.

Traceability might be precondition for the acceptance of a process in a company. The clear relationship between the derived process and the original standard provides support for arguing process-compliance. Every user of the process is able to understand which section of a standard is base for the definition of a derived process element. For this reason a direct link to the standard is part of every process element beneficial during a process audit.

Modelling limitations. As pointed out by predecessors and as also found out through this work, SPEM2.0/EPF Composer offers a limited variability modelling support, which makes it not ideal for modelling a process line.

Flexible use of the notion of partial commonality resulted to be a strategic solution for increasing the identification of common process elements.

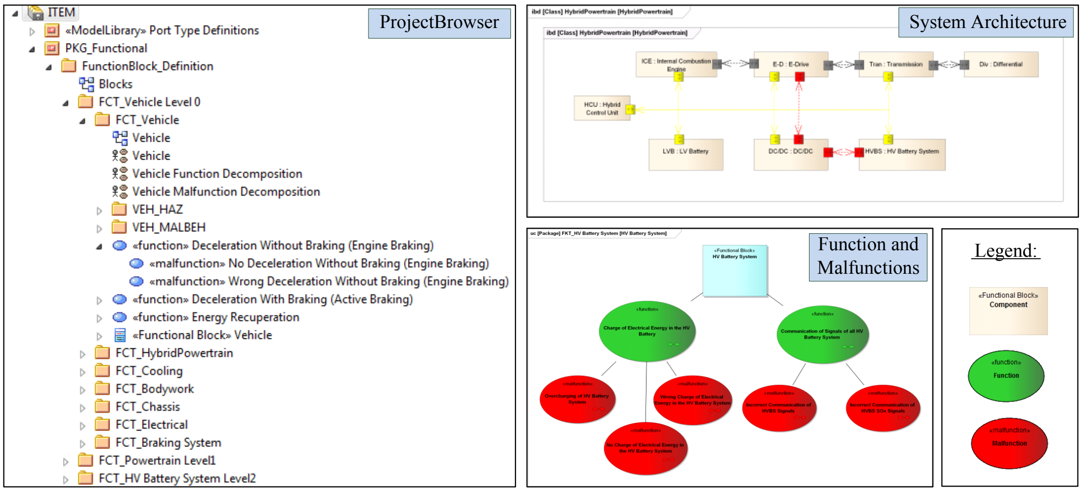

ProjectBrowser
ITEM
«ModelLibrary» Port Type Definitions
PKG_Functional
FunctionBlock_Definition
Blocks
FCT_Vehicle Level 0
FCT_Vehicle
Vehicle
Vehicle
Vehicle Function Decomposition
Vehicle Malfunction Decomposition
VEH_HAZ
VEH_MALBEH
«function» Deceleration Without Braking (Engine Braking)
«malfunction» No Deceleration Without Braking (Engine Braking)
«malfunction» Wrong Deceleration Without Braking (Engine Braking)
«function» Deceleration With Braking (Active Braking)
«function» Energy Recuperation
«Functional Block» Vehicle
FCT_HybridPowertrain
FCT_Cooling
FCT_Bodywork
FCT_Chassis
FCT_Electrical
FCT_Braking System
FCT_Powertrain Level1
FCT_HV Battery System Level2
System Architecture
ibd [Class] HybridPowertrain [HybridPowertrain]
ibd [Class] HybridPowertrain [HybridPowertrain]
ICE : Internal Combustion Engine
E-D : E-Drive
Tran : Transmission
Div : Differential
HCU : Hybrid Control Unit
LVB : LV Battery
DC/DC : DC/DC
HVBS : HV Battery System
uc [Package] FKT_HV Battery System [HV Battery System]
Function and Malfunctions
«Functional Block» HV Battery System
«function» Charge of Electrical Energy in the HV Battery
«function» Communication of Signals of all HV Battery System
«malfunction» Overcharging of HV Battery System
«malfunction» Wrong Charge of Electrical Energy in the HV Battery System
«malfunction» No Charge of Electrical Energy in the HV Battery System
«malfunction» Incorrect Communication of HVBS Signals
«malfunction» Incorrect Communication of HVBS SOx Signals
Legend:
«Functional Block» Component
«function» Function
«malfunction» Malfunction

Figure 4.3: Demonstration of system model in use case scenario 2 [Tool: EA]

4.3 Use Case Scenario 2: Modelling, Analysis and Simulation of Safety Artefacts

The use case scenario covers following aspects:

- Demonstration of Methodology 2
- Domain: Automotive Domain
- Relevant Standards: Safety - ISO 26262
- Integrated system modelling approach that supports safety analysis and early verification by simulation

4.3.1 Description Use Case Scenario 2

System Model. The system model covers all kinds of interactions and influences between the battery system and the vehicle behavior based on SysML. The model shows connections between components, functions, vehicle malfunctions, powertrain and battery system level. Each component is represented by a functional block, and the functions and malfunctions are represented by use case elements. Figure 4.3 shows the implementation in the system modelling by EA tool (project browser, system architecture and connections between components, function and malfunction). Based on the single source principle, relevant safety artifacts are represented in the system model. Each safety artifact is modeled with SysML modeling elements (e.g. preliminary architecture) or a link to an external safety work product (e.g. item definition can be linked in MS Word). Thus, the system model model represents traceability between different safety artifacts.

Interaction of System Model and Safety Analysis. One aim of the system model is the support of creation of the three main data nets for the safety analysis e.g. to perform FMEA: structure net, function net and failure net. For the execution of the safety analysis, the required safety artifacts are transferred from EA to the external safety analysis tool. After performing the safety analysis, the resulting data are transferred back to the system model. The FMEA is used to derive suitable safety measures to support the elaboration of the functional safety concept for the battery system. For the FMEA, the APIS IQ FMEA tool was used, because it is an established tool for reliability analysis by quality engineers in the automotive industry. Furthermore, this tool can be extended to be used for functional safety analysis aspects. Thus, it is possible to combine the quality and safety analyses because the same expert team is involved in the analysis activities (Figure 4.4). More details about the evaluation of the MBSE approach for HV battery systems regarding the interaction of system model and safety analysis are described in [81].

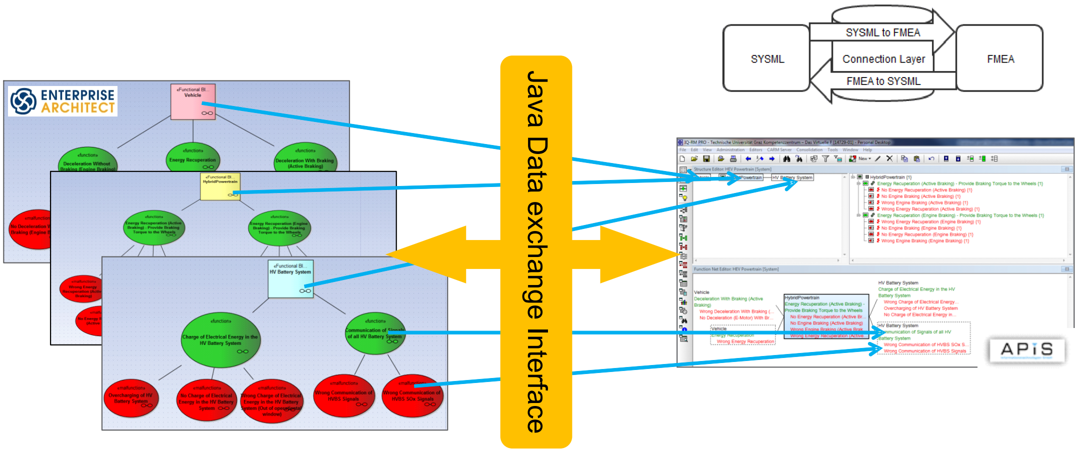


Figure 4.4: Demonstration of system model interaction with FMEA in use case scenario 2 [Tools: EA, SafetyDataExchange, APIS FMEA]

Interaction of System Model and Simulation. Several strategies are known to charge and discharge an automotive battery propelling a vehicle. Information about driver behavior, routing, road profile, etc. have a strong influence on the behavior of the battery system. Such a battery system usually consists of two main components, namely the battery pack as energy storage facility and a corresponding battery controller. The targeted battery pack consists of, e.g. 4 battery modules, where each of them integrates 12 cells with a nominal capacity of 24Ah each. The targeted controller monitors operational condition and health of each of the modules. The controller also communicates with the hybrid control unit of the vehicle to ensure a stable vehicle operation. For the battery module, a SystemC-AMS based FMU utilizing the electrical linear network and timed dataflow models of computation (MoC). The battery controller is implemented in a SystemC/SystemC-AMS based FMU utilizing the discrete event and linear signal flow MoC. Both FMUs are integrated into one common co-simulation scenario. More details about the used simulation models can be found in [82]. Figure 4.5 shows the results of the simulation. From a qualitative point of view, the battery simulation results correspond to the results described in [89]. Quantitatively, the generated battery module FMU reproduces the simulation results shown in [90], validating against Li-Po cells from [49]. For this case no relevant increase of simulation time caused by the co-simulation framework was measured. More details about the evaluation of the MBSE approach for HV battery systems regarding the interaction of system model and simulation are described [82].

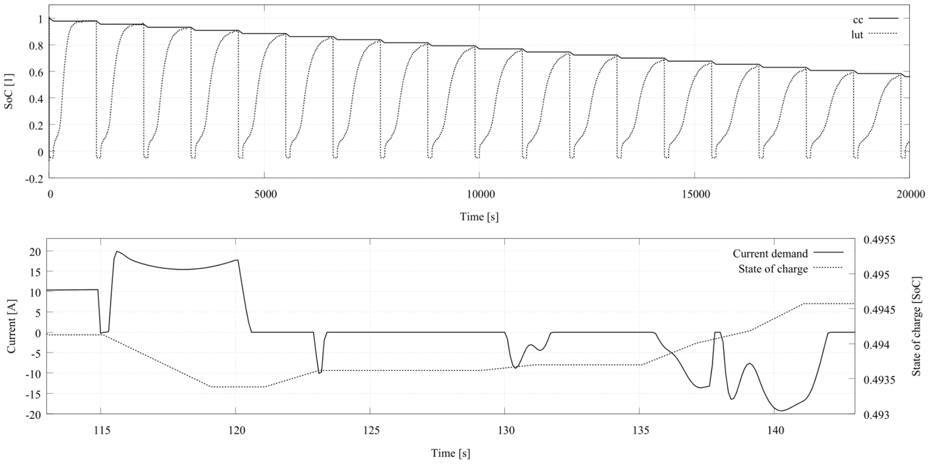


Figure 4.5: Demonstration of HV battery simulation in use case scenario 2 [Illustration: MATLAB] [82]

4.3.2 Results of Use Case Scenario 2

Common source of safety artifact data. The model-based safety engineering approach provides support for safety activities in the functional safety process as defined by ISO 26262. The approach supports a systematic description and analysis of different kinds of safety artifacts in a common system model.

Traceability within safety artifacts. All functional aspects are represented, including coverage of the dependencies between different kinds of safety artifacts.

Data exchange between model and safety analysis. Bi-directional data exchange between system modeling tool and functional safety analysis tools; Export of relevant artifacts for specific safety analysis methods and import of preventive and detection actions as a basis for deriving safety requirements.

4.3.3 Lessons Learnt of Use Case Scenario 2

Definition of complete safety modeling profile. The profile mechanism of UML/SysML should be investigated for the definition of a safety modeling profile that defines a reduced subset of modeling elements and their associations for the whole safety lifecycle.

Check of safety modeling artifacts. Based on the defined safety modeling profile, the created artifacts of a specific safety activity could be checked to see if they are modelled according to the profile and if all required traceability links to related artifacts are present.

Simulation by using FMU. The creation of the required FMU XML file and synchronization to the model code causes additional efforts due to variable numbering and name assignments, especially if models are modified. Additional automation by use of a MBSE development approach is beneficial.

4.4 Use Case Scenario 3: Application of Safety and Security Co-Engineering and Argumentation Framework

The use case scenario covers following aspects:

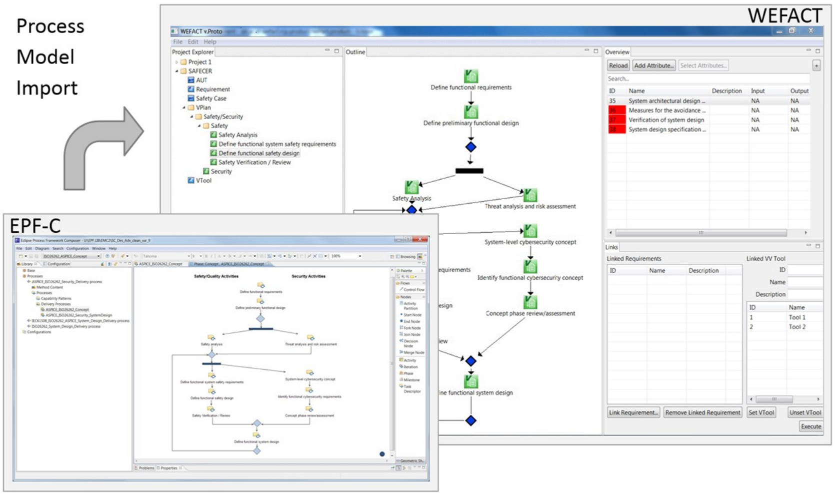
- Demonstration of Methodology 3
- Domain: Automotive Domain
- Relevant Standards: GSN Community Standard (Argumentation), ASPICE (Quality), ISO 26262 (Safety), SAE J3061 (Security)
- Framework provides co-engineering of safety and security process modelling, process execution, co-analysis and argumentation

4.4.1 Description Use Case Scenario 3

For safety and security it is required to provide evidence and argumentation to show that system development was done compliant to relevant standards and that the system satisfies safety and security goals. The final documentation has to be provided by the assurance case including safety and security.

Process Definition and Process Execution. Efficient safety certification implies a process model which guides the user through the certification process and allows efficient compositional re-certification in the event of changes in the system. EPF-C provides elements to model phases and individual activities of the safety and security process. It allows modelling specific standards in a formal way, which enables automating the certification workflow.

Safety and Security Co-Analysis using STPA-Sec. The identification of potential safety-related accidents based on potential causes regarding safety and security have been supported by the tool XSTAMPP, e.g. failures and malicious manipulations by an attacker. In an independent analysis the focus of security would be on the classical CIA properties (confidentiality, integrity and availability). The feedback of safety relevance of these certain properties is missing. Safety specific analysis focuses only on safety issues caused by faults of E/E systems. Scenarios in which a user modifies the vehicle and causes a potential safety hazard would be missed. Co-Analysis connects the domains and supports the identification of safety goals and safety-related security goals.


Process
Model
Import
WEFACT
WEFACT v.Proto
File Edit Help
Project Explorer
Project 1
SAFECER
AUT
Requirement
Safety Case
VPlan
Safety/Security
Safety
Safety Analysis
Define functional system safety requirements
Define functional safety design
Safety Verification / Review
Security
VTool
Outline
Define functional requirements
Define preliminary functional design
Safety Analysis
Threat analysis and risk assessment
System-level cybersecurity concept
Identify functional cybersecurity concept
Concept phase review/assessment
ine functional system design
Overview
Reload
Add Attribute...
Select Attributes...
Search...
Links
Linked Requirements
Linked VV Tool
Link Requirement...
Remove Linked Requirement
Set VTool
Unset VTool
Execute
EPF-C
Eclipse Process Framework Composer
Safety/Quality Activities
Security Activities

ID	Name	Description	Input	Output
35	System architectural design ...		NA	NA
36	Measures for the avoidance ...		NA	NA
37	Verification of system design		NA	NA
38	System design specification ...		NA	NA

ID	Name
1	Tool 1
2	Tool 2

Figure 4.6: Process Model transfer in Use Case Scenario 3 [Tools: EPF-C, WEFACT] [86]

Patterns for Process and Argumentation. Patterns are used to provide process and argumentation frameworks, which represents most of the recurring steps. The intention is to spend time once and reuse the elaborated patterns many times. Especially the integration of activities related to functional safety and security is a challenging work. The created patterns provide process- and argumentation-templates. Process patterns simplify creating development processes because they already bring together functional safety and security activities. Argumentation patterns are corresponding to the process and exhibit the line of argumentation using the created work products. They include argumentation concerning safety and security and the interaction between them. Both types of patterns have to be instantiated for the specific development project. Instantiation for example means to select project specific methods like STPA-Sec for co-analysis. In parallel, the corresponding line of argumentation has to be selected. The purpose of creating patterns within the framework is to simplify the process definition, where the elaboration of evidence and adequate fitting arguments supports claims related to requirements.

Product-based Argumentation. One result of the co-analysis is the idenfication of the malfunction "overcharging battery during plug-in charging" for which developers have to implement an adequate counter measure. Overcharging will be possible if an attacker modifies the BMU parameters. To document the relationship between requirements (represented as goals) and measures (declared in evidence documents) the tool OpenCert is used for argumentation by GSN. On the one hand the argumentation covers the safety and security process and on the other hand it deals with the product specific decision how to prevent "Battery overcharging". From the security process point of view the top level claim is "Define functional cybersecurity requirements to prevent unauthorized changes to BMU parameters". These requirements are listed in the corresponding project specific document "HV Batt SecReq" stored in the project repository. From the product point of view the BMU needs capabilities to detect and prevent unauthorized change of parameters. The documentation of these capabilities is evidence and usable as product-based

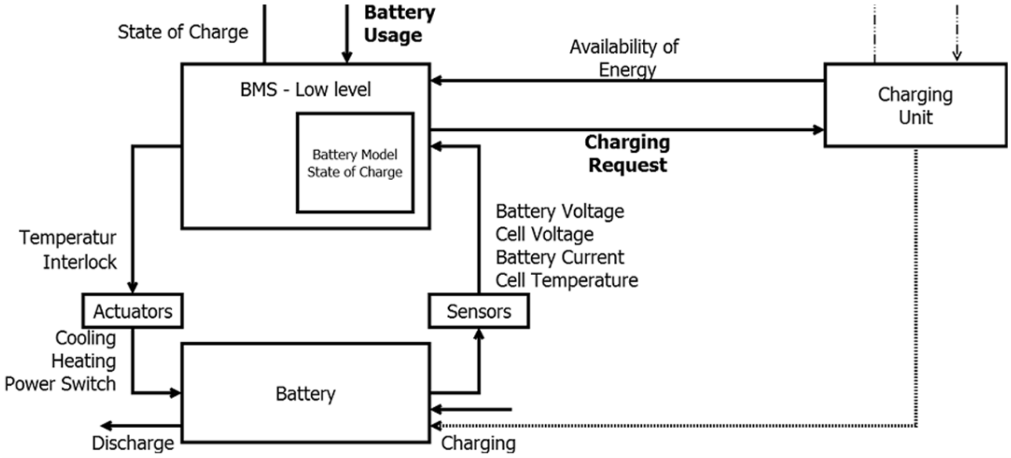


Figure 4.7: HV battery system co-analysis model [Tool: XSTAMPP] [86]

argumentation.

More details can be found in the publication [86].

4.4.2 Results and Lessons Learnt of Use Case Scenario 3

Interaction between safety and security. The application of the Methodology 3 showed a possible way how functional safety and security should correspond. Interaction between safety and security was forced by additional joint activities.

Safety and Security Co-Analysis. The co-analysis method STPA-Sec was used and supported by the tool XSTAMPP. Product specific safety and security measures were coordinated to prevent unwanted interaction.

Argumentation Patterns. The graphical depiction of links between these elements improves the stakeholder's understanding and shows how the dependencies between safety and security are organized. The execution of the assurance activities by the workflow engine (WEFACT) supported generation of evidences for the combined safety and security case. The tool OpenCert provides the possibility to manage patterns and to create GSN structures. The usage of patterns speeded up the process definition activities and supported creation of argumentation fragments by GSN, which connect processes and evidence with argumentation.

4.5 Research Findings

Process modelling allows more interactive work with the standard, reduces time and effort for engineers to find relevant paragraphs and to comprehend them as a company-wide interpretation. However, converting a standard might introduce differences due to human errors to the original standard, which could in the worst case lead to wrong interpretations during development. Nevertheless, by using the process model the needed sections within the standard can be found faster and activities can be performed based in a common understanding.

Model development for industry sized problems is time consuming and requires new skills in the industry. The model-based system development approaches and the implemented tool chain allows the seamless data exchange between different safety analysis tools. Furthermore, traceability and consistency between different work products can be achieved during the development. The presented implementation of the approach has been evaluated on several industrial use cases, where the practical applicability could be acknowledged.

The required level of detail of the system model needs to be defined before modelling starts, because it impacts the expressiveness of the tests used during early validation. On the one hand, if the modelling details are too low, the tests cannot ensure that a certain safety measure is sufficient. On the other hand, if the modelling detail is too high, the creation of the needed models and tests is too time consuming, limiting its applicability in industrial applications.

The usage of GSN for graphically description of the safety argument also supports engineers in finding and understanding the arguments relevant for their work during the development and for the safety assessment. Since the usage of a model-based approach of the argumentation does not change its content, the overall achieved safety by the chosen safety measures is the same in comparison to common, document bases approaches.

The investigations performed regarding the safety and security co-engineering framework show that these two topics are heavily interleaved and need to be addressed together. The proposed co-engineering framework outlines the needed items and provides the basis for the implementation in an industrial setting for functions which have to be safe and secure.

4.6 Application of MBS^{2}E approach for Automated Driving

Investigations have been done regarding the relevance of Functional Safety according ISO 26262 for automated driving functions together with industrial experts from OEM (VOLVO) and Supplier (MAGNA Steyr Fahrzeugtechnik) [91]. Furthermore, there are still on-going investigations regarding the evaluation of the presented approach for the topic of automated driving, but the results have not been published so far. First results are promising but specific adaptations have to be done to cover the different automation levels without human response regarding the controllability and the hand-over concept from Automated Driving Assistance Systems (ADAS) to human driver.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

The thesis at hand addresses existing challenges of the development of safety-critical systems regarding (a) different engineering standards (e.g. quality, safety, security) have to be considered in an integrated process model, (b) a central system model that contains all relevant artefacts to support safety analysis and simulation for early validation, (c) safety argumentation that covers the demands of ISO 26262 safety case generation collaterally to the engineering lifecycle, (d) approaches for reuse of safety artefacts, and (e) interaction of safety and security aspects.

For that reason this work investigates four main Model-Based System Safety Engineering (MBS^{2}E) methodologies, that cover the following aspects:
(A) An approach for Process Modelling and Management to reuse different kind of process elements, derive a project specific process model and execute this process model during project execution to track required activities and artefacts. (B) Modelling, Analysis and Simulation of E/E Systems provides support to gather relevant system artefacts in a central system model by introducing a safety-extension for system modelling based on SysML. The safety elements in that model are iteratively extended with the results of safety analysis activities. Furthermore, the structure and parameters of the system model are used for configuration of the simulation model, which is used for early safety-validation implemented in a SystemC environment. (C) Process- and Product-based Argumentation part uses a semi-formal modelling language called Goal Structuring Notation, where preexisting argumentation patterns for process- or product-based argumentation are collected and used as best practices for the specific project needs. (D) The Safety and Security Co-engineering Framework provides a possible approach to guide engineers through development steps compliant to specific standards. The relevant interaction points of safety and security activities are considered in an iterative and systematic way.

The applicability of the presented MBS^{2}E methodologies are demonstrated on an automotive high voltage battery system use case within a hybrid powertrain. These methodologies are applied to three specific use case scenarios regarding (1) process modelling and management, (2) safety artefact modelling to support safety analysis and simulation, and (3) safety and security co-engineering and argumentation. The demonstration of the elaborated methods provides a proof of concept showing that identified improvements have been covered to a specific extend. The use case scenarios show how to thoroughly engineer a modelling framework supporting flexible process model definition and thus allowing process engineers to select and compose process elements in compliance with required standards. Modelling of safety artefacts is a feasible approach for derivation of safety requirements supported via a system model in SysML. The performance of various safety analysis methods supported by data exchange based on one system model centralizes the data and avoids inconsistency. Early verification and validation activities, enabled by extending the system model with parameters to support standardized SystemC or SystemC-AMS, libraries eases the integration of existing system level simulation models into larger and more complex simulation scenarios, which are used for information exchange on system level. The safety and security co-engineering framework provides the possibility to perform co-analysis that considers potential negative influence and derives harmonized safety and security measures in an early development stage.

5.2 Future Work

Further investigation based on the presented MBS^{2}E approach for other future automotive applications like automated driving functions are work in progress. The ISO 26262 provides a basis regarding automotive functional safety for handling hazards caused by malfunctioning behaviour of E/E safety related systems. This ISO standard is also applicable to any level of automated driving, but the system complexity challenges engineers today, because of the higher degree of networking functionalities that must be handled, which has been investigated together with industrial partners from VOLVO and MAGNA Steyr Fahrzeugtechnik [91]. For that reason further aspects must be considered to realise automated driving functions in an adequate manner (e.g. availability, reliability, safety, and security).

Automotive standardization regarding functional safety develops further and there are three main topics that have to be addressed in future investigations based on the presented MBS^{2}E approach:

- Upcoming changes in 2nd Edition of ISO 26262
- Safety of the intended Functionality (SOTIF)
- Automotive Cybersecurity

Upcoming changes in 2nd Edition of ISO 26262. International standards like ISO 26262 requires updates, which are discussed after a specific period of time after its publication allowing industry to gain experience.

At the moment the ISO 26262 is in final revision phase and it is planned to release the 2nd edition of ISO 26262 by Q4/2018. The second edition will provide changes regarding the scope of application and further two new parts: The scope will be extended from cars to other road vehicle such as motorcycles (excluding mopeds), trucks, busses, trailers and semitrailers. That means that ISO 26262 is also valid for road vehicles>3500kg. For trucks and busses there are additional notes and annexes available marked with "T&B"

in the specific sections of part1/2/3/4/7/8. Two new parts will be introduced, Part 11 - "Guideline on application of ISO 26262 to semiconductors" and Part 12 - "Adaptation for motorcycles". Part 11 will include semiconductor components and its partitioning, specific semiconductor technologies and use cases and further examples for evaluation of safety mechanisms, dependent failure analysis and quantitative analysis. Part 12 will specify the adaptations for motorcycle regarding safety management during the concept phase and the product development, hazard analysis and risk assessment, vehicle integration and testing, and safety validation. The upcoming version of ISO 26262 will not address cybersecurity directly, but it will provide an informative annex "Guidance on potential interaction of functional safety with cybersecurity" where possible interaction points are highlighted in different lifecycle phases. Furthermore, some parts provide additional notes where an interaction may influence the functional safety aspects. The mentioned changes and extensions have to be taken into consideration for an update of the presented approach in that thesis, which may need some further extensions to cover that aspects.

Safety of the intended Functionality (SOTIF). The scope of ISO 26262 is intended to functional faults in the E/E system and covers random hardware faults and systematic faults in the development, but the prevention of any safety issues based on the functional insufficiency in particular of automated driving systems are out of scope. These systems can lead to safety violations if a hazardous decision is made by the processing algorithm (e.g. about the environment), which is based on sensor inputs, even in absence of a fault in the system. The topic of SOTIF will bring development of nominal performance and safety engineering closer together. Any effect or interaction during product development have to be considered over entire product lifecycle. For that reason the functional safety working group decided to start with the elaboration of a Public Available Specification (PAS) ISO/PAS 21448 "Road vehicles — Safety of the Intended Functionality" that is planned to release by Q4/2018. The presented approaches in that thesis could be extended to cover SOTIF as a further standard.

Automotive Cybersecurity. The interconnection and intercommunication of functions brings the topic of cybersecurity at the table. The elaboration of an international automotive cybersecurity standard has been started ISO/SAE AWI 21434 "Road Vehicles - Cybersecurity engineering" in an joint working group of experts from ISO and SAE[1]. Many discussion are going on at the moment to elaborate a common base terms and definitions, different analysis methods along the engineering lifecycle, and harmonisation points between safety and security and how to handle the trade-off between different attributes and derived counter measures. Furthermore, it comes to a change from development to engineering lifecycle, because the engineering is not finished after SOP. The phases after production (post production) become more important, because OEM and suppliers have to monitor and support their products more intensive during operation and maintenance. It is planned to have the final international standard regarding cybersecurity available by Q1/2020.

The presented approach in that thesis could be adapted for the needs of cybersecurity,

[1] The guidebook of SAE J3061 will be replaced by the joint the upcoming ISO/SAE standard.

because a similar approach as given in ISO 26262 will be used in the upcoming cybersecurity standard. The different interaction points of functional safety and cybersecurity have to be communicated clearly to provide a secure and safe product to customer market. As a next step, the extension to Model-based System Safety and Security Engineering will be a promising approach to fulfil that needs.

Chapter 6

Publications

This chapter contains all publications by the author of this thesis, which explain the presented approach in Chapter 3 and use case scenarios presented in Chapter 4 in more detail.
Fig. 6.1 shows the mapping between the individual publications and the proposed methodology for the model-based safety engineering for automotive systems.

The methodology as shown in Fig. 6.1 covers the following safety engineering topics:

- **Process Modelling & Management**
  (See section 3.1 for more details).
- **Model-based Safety Engineering Modelling, Analysis and Simulation**
  (See section 3.2 for more details).
- **Support for Safety Argumentation and Safety/Security Co-Engineering**
  (See section 3.3 for more details).

The following relevant industrial use case is used for demonstration and evaluation of the methodology:

- **Hybrid Electric Vehicle Powertrain with focus on HV Battery System**
  (See section 4.1 for more details).

6.1 Summary of appended Publications

The following publications represent the main contribution of the thesis at hand. For each publication a short summary is given and the contribution of the author is highlighted. In the end of this chapter all publications are appended with their full text.

Publication A (2014_SAE): *Challenges for reuse in a safety-critical context: A state-of-practice study [77]*, 2014 SAE World Congress, Detroit, Michigan, USA, April 8-10, 2014
Content: The paper shows the results of an explorative study and provides an overview

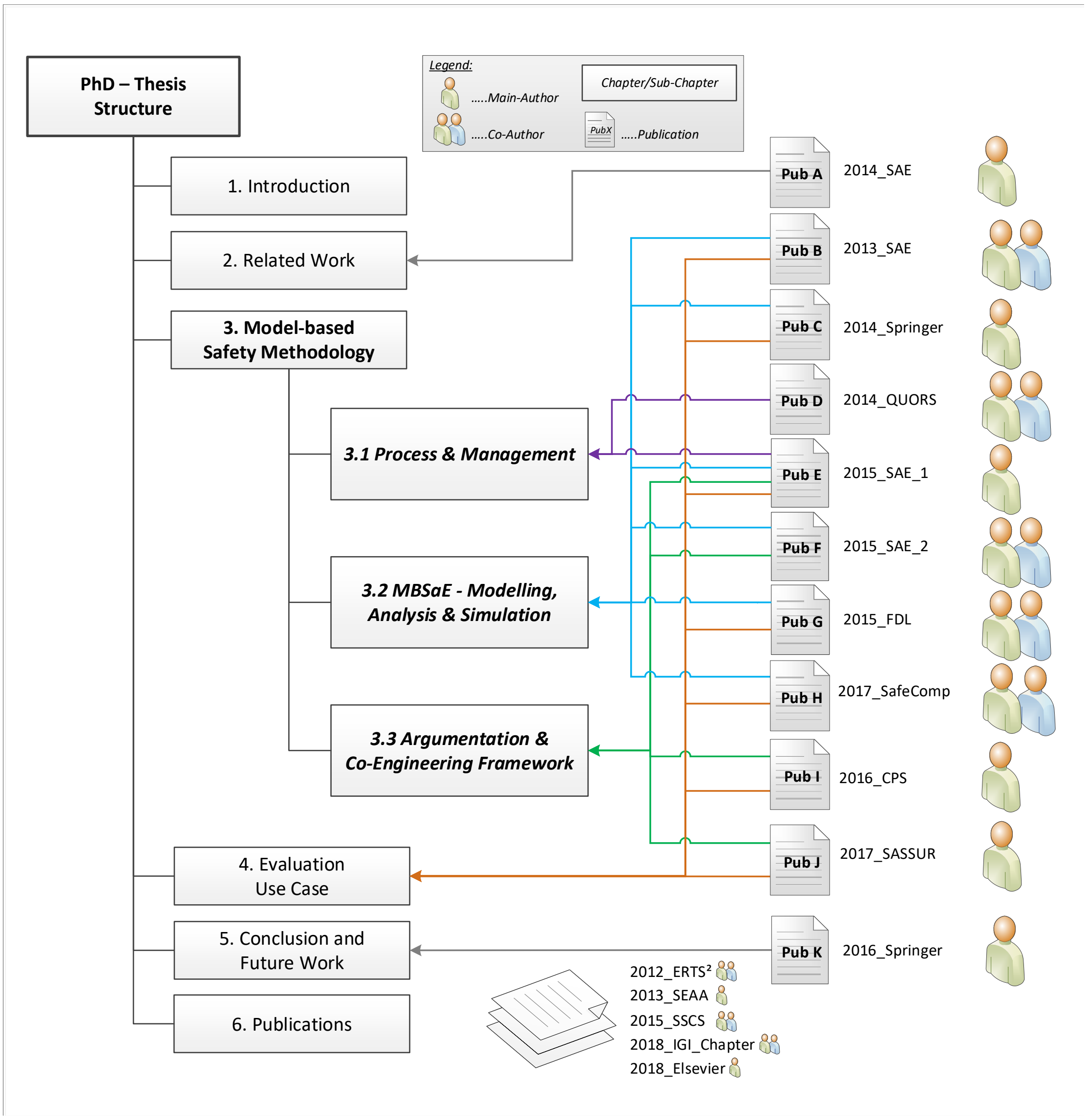


Figure 6.1: Publication Overview of PhD Thesis

of current state of practice concerning reuse in context of functional safety. Based on the results of this survey the main challenges have been derived and suggested possible improvements as future work to overcome these challenges.
Contribution of author: Main author of all parts.

Publication B (2013_SAE): *System Modelling for Integration and Test of Safety-Critical Automotive Embedded Systems [92]*, 2013 SAE World Congress, Detroit, Michigan, USA, April 16-18, 2013
Content: The paper proposes a system modelling based approach for integration and test of safety-critical embedded systems. An extension of the V-Model targets process oriented needs for safety and indicates, where modelling languages in favour can be applied best.

To establish a link between safety goals and the structure of simulation models, the initial model is enriched with necessary information and transformed to a language suitable for advanced simulation tasks. The proposed system modelling based approach enables safety verification and validation at an early stage of development.
Contribution of author: Elaboration of joint methodology for modelling and simulation.

Publication C (2014_Springer): *Holistic Safety Considerations for Automotive Battery Systems [88]*, Automotive Battery Technology. Springer International Publishing, 2014
Content: This paper proposes a workflow for elaboration of an integrated safety concept including safety measures from different engineering disciplines. Two main lessons learned are that the consideration of all kinds of risk reduction measures in the concept phase improve the understanding of safety of the overall system, and involving various fields of expertise enables the development of a clear safety concept. This approach improves the development of the overall system, while complying it with requirements of ISO 26262 for the development of E/E systems. The applicability of the introduced approach is demonstrated on an automotive battery case study.
Contribution of author: Main author of all parts.

Publication D (2014_QUORS): *Modelling a Safety and automotive oriented Process Line to enable Reuse and Flexible Process Derivation [78]*, 8th IEEE International Workshop Quality-Oriented Reuse of Software (QUORS‘14), 2014
Content: This paper proposes a methodological framework for implementing the safety-oriented process line approach. More specifically, we have examined three standards that are used in the automotive domain and after having identified commonalities and variabilities we have shown how to systematically model them in SPEM2.0/EPF. We have also shown how those commonalities and variabilities can be exploited for the definition of flexible processes. From this work we have drawn some lessons learned: the examined processes exhibit commonalities and thus the safety-oriented process line approach represents a sound and effective way for systematizing reuse and enabling the introduction of changes that might be required when switching from one standard to another (e.g. for intra-domain re-certification). The current tool support for modeling is quite limited.
Contribution of author: Extending and application of the SoPL approach for an automotive use case.

Publication E (2015_SAE_1): *Model-based Engineering Workflow for Automotive Safety Concepts [81]*, 2015 SAE World Congress, Detroit, Michigan, USA, April 21-23, 2015
Content: The paper proposes a safety workflow that covers the concept phase of the automotive functional safety standard ISO 26262. The approach provides an overall view of relevant semi-formal safety modeling artifacts. For the modeling of safety artifacts, a SysML profile has been introduced. It supports creation and management of safety artifacts required in the safety workflow. An integrated safety analysis tool chain was demonstrated using the Enterprise Architect system modeling tool and APIS IQ FMEA

for the Concept FMEA. The applicability of the approach was demonstrated using an automotive use case of a battery system of an HEV powertrain, which showed that the approach is generally suitable for enhancing the quality of artifacts in the safety workflow and the safety argumentation. The combination of a safety-oriented workflow and semi-formal modelling helps relevant stakeholders perform safety engineering activities in a systematic way, as required by various standards such as ISO 26262.
Contribution of author: Main author of all parts.

Publication F (2015_SAE_2): *From Natural Language to Semi-Formal Notation Requirements for Automotive Safety [93]*, 2015 SAE World Congress, Detroit, Michigan, USA, April 21-23, 2015
Content: Two key points can be observed in course of this work. One is that the ISO 26262 standard would be further advanced with a clear and concise work product to define a state of the art safety requirement. A product suffers greatly without having a specific template to check technical and process compliance of the conversion of the natural language requirements into semi-formal notation to avoid systematic failures to aid model based development. Several developers with different perspectives automatically introduce these systematic failures. Second is that without a known method how to verify the entire range of faults (systematic fault and random faults) with a 100% branch coverage are not very effective or beneficial. The complexity of today's automotive functionality supports the requirements of the ISO 26262 standard to shift to model based engineering. The selected tooling must be both consistent and complete, to cover those faults which could cause harm at the item level.
Contribution of the author: Support the author for safety and reuse aspects of requirements patterns.

Publication G (2015_FDL): *Standard Compliant Co-Simulation Models for Verification of Automotive Embedded Systems [82]*, 2015 Forum on Design and Specification Languages (FDL), Barcelona, Spain, September 14-16, 2015
Content: In this paper a structured method for the integration of SystemC/SystemC-AMS simulation models to the FMI standard is introduced. The presented method does not require any changes to the standardized SystemC or SystemC-AMS libraries. The method eases the integration of existing system level simulation models into larger and more complex simulation scenarios, which are used for information exchange and verification on system level. A two-part battery system use case from the automotive domain is presented, which exploits these models of computation for simulation. The resulting FMUs, created with the described method, are highly transportable and configurable. These properties make them suitable for verification and information exchange processes within the automotive domain.
Contribution of the author: Elaboration of requirements and a methodology related to co-simulation data-exchange, modelling and integration.

Publication H (2017_SafeComp): *Systematic Pattern Approach for Safety and Security Co-Engineering in the Automotive Domain [94].* SAFECOMP2017
Content: This paper focuses on the selection, combination, and application of safety and

security patterns. The introduction of the pattern engineering lifecycle provided a systematic way of safety- and security-related pattern engineering process steps to development, and included already existing work products, such as the results of safety analyses. Safety and Security Co-Engineering Loops helped to align these activities systematically. It benefits from tight integration of safety- and security-related process steps, which requires increased exchange of information between them. An industrial use case demonstrated the practical realization of our approach. With the presented approach, we aimed to derive the manifold benefits from patterns inherent to their nature. This is a mean for accelerating the application of adequate safety and security co-engineering in the automotive domain. In particular, we showed a way to remediate the lack of security knowledge and facilitate easier and more informed integration of these two separate yet interfering disciplines.
Contribution of author: Joint elaboration of the methodology and exemplary application use scenario.

Publication I (2016_CPS): *Process- and Product-based Lines of Argument for Automotive Safety Cases [83]* EMC2 Summit. CPS Week. 2016
Content: This paper presents a methodology to create argumentation structures which are in direct relation to development processes and demanded requirements. The formalism deals with patterns and templates to make it easier to establish a complete understandable line of argumentation and prevents information loss. A workflow has been defined to introduce a methodology for process- and product-based argumentation. Project specific tailoring is used to create a standard compliant development process. Instantiation of templates provided by the engineering process leads to product specific safety argumentation. Application of the proposed workflow results in a complete and structured safety argumentation, which is needed for the safety case and supports functional safety audits and functional safety assessments. First experiences have been gained by successful application to an automotive battery use case to ensure compliance with ASPICE and ISO 26262.
Contribution of the author: Main author of all parts.

Publication J (2017_SASSUR): *Safety and Security Co-Engineering and Argumentation Framework [86]* SAFECOMP2017 - Workshop SASSUR
Content: Today's interconnected world needs special care to consider safety and security aspects. Although there are approaches treating the interaction between safety and security adequately, they are still immature. This paper presented a safety and security co-engineering framework. A comprehensive combined safety and security argumentation methodology for the automotive domain has been developed. Its application in the automotive domain within these standards constraints provides useful information and can be considered as the next step for a wide application in development lifecycles. The following important benefits of the presented methodology for argumentation apply to the automotive domain: Usage of patterns improves process definition; the GSN structures connect process- and product-related evidence with argumentation; the graphical depiction of links between elements improves the stakeholder's understanding of relevant safety and security aspects. In the HEV powertrain use case we showed the benefit of combined analysis of

safety and security issues and the preparation of an assurance case for safety and security.
Contribution of author: Main author of all parts.

Publication K (2016_SPRINGER): *Functional Safety of Automated Driving Systems: Does ISO 26262 Meet the Challenges? [91]*, Automated Driving Safer and more efficient future driving. Springer International Publishing, 2016
Content: Different kinds of challenges must be considered to realise Automated Driving System (ADS) functions in an adequate manner. Following challenges are discussed in this paper: Increasing complexity of highly interconnected functions and influence of system attributes, such as availability, reliability, safety, and security must be harmonised. The concept phase of ISO 26262 becomes more important for ADS functions because development of ADS requires engineering approaches and technologies beyond state of the art. In particular, influence of the driver in the HARA, definition of safety goals and corresponding attributes for specific levels of ADS (e.g. safe state) as well as changes of the functional safety concept from fail-safe to fail-operational strategies. Today, several methods are available to support complex systems but they must be improved for the development of ADS. Possible technologies are discussed to handle the increasing complexity: Model-Based Systems Engineering, formal verification by contract based development, as well as simulation and co-simulation.
Contribution of author: Main author of all parts.

Further involved publications of the author are mentioned here and not included as full-text in this chapter:

Publication (2012_ERTS^2): *Using the CESAR Safety Framework for Functional Safety Management in the context of ISO 26262 [95]*, 2012 Embedded Real Time Software and Systems, ERTS^{2}12, Toulouse, France, 1st - 3rd February 2012
Content: Functional safety management in the context of ISO 26262 is a challenging task due to the amount of activities and large number of requirements listed in the standard, as well as the size of the distributed development teams over a number of organizations involved in an engineering project. The availability of the CESAR safety framework as knowledge data base is very useful for the systematic planning of safety activities required in context of ISO 26262. This paper shows how this information can be used in an industrial context and how the tailoring for a company (which require additional company internal information is required) can be performed. The resulting framework has been already used in several customer projects and was a central brick in order to synchronize development activities between different partners and finally for the success of the projects.

Publication (2013_SEAA): *Investigation of the influence of non-E/E safety measures for the ASIL determination [96]*, 39th Euromicro Conference on Software Engineering and Advanced Application, Santander, Spain, September 4-9, 2013
Content: This paper summarizes the main findings during the investigation concerning functional safety for high voltage batteries typically used in electric or hybrid electric vehicles. An approach for an iterative determination of the required ASIL by applying non-E/E measures is presented. We observe that it is often meaningful to consider external measures and other technologies early in the concept phase. The incorporation

of different engineering disciplines with different viewpoints helps to improve the system safety concept. Only considering E/E measures would result in much higher ASIL requirements and therefore increase the safety case development effort and the complexity of safety argumentation. We show that we can considerably reduce the required ASIL for some hazards by considering non-E/E measures early in development.

Publication (2015_SSCS): *A Framework for Model-Based Safety Requirements Roundtrip Engineering [97]*, 10th IET System Safety and Cyber Security Conference, The Institution of Engineering and Technology (IET), Bristol, UK, October 20-22, 2015
Content: In this paper, a software tool for automotive safety engineering called AVL Safety Extensions (AVL-SE) is presented. As part of a tool framework, AVL-SE supports a system safety engineering workflow aligned with ISO 26262-3 and -4 and requires the use of a SysML-based modelling language. The software tool framework contains a database for a system model and a database for requirements both containing safety requirements. AVL-SE allows automatic synchronisation of databases in terms of requirements, traces and allocations. This enables full safety requirements round-trip engineering using a tool for SysML-based modelling and a tool for requirements engineering. Workflow and tool were experimentally applied in the case study of an HEV powertrain E/E system. It has been appeared that the presented approach to round-trip engineering effectively eases the handling of the vast number of requirements which emerge from engineering an HEV powertrain E/E system.

Publication (2018_IGI_Chapter): *Integration of Security in the Development Lifecycle of Dependable Automotive CPS [10]*, Chapter in Handbook of Research on Solutions for Cyber-Physical Systems Ubiquity. IGI Global 2018
Content: With the introduction of connected vehicles, the automotive domain must now consider cybersecurity as an integral part of the development lifecycle. Just as safety became a critical part of the development in the late 20th century, modern vehicles are required to become resilient against cyberattacks. The exciting new features, such as advanced driver assistance systems, fleet management systems, and autonomous driving, drive the need for built-in security solutions and architectural designs to mitigate emerging security threats. Thus, cybersecurity joins reliability and safety as a cornerstone for success in the automotive industry. As vehicle providers gear up for cybersecurity challenges, they can capitalize on experiences from many other domains, but nevertheless must face several unique challenges. This article thus focuses on the enhancement of state-of-the-art development lifecycle for automotive cyber-physical systems toward the integration of security, safety and reliability engineering methods. Four engineering approaches in particular (HARA at concept level, FMEA and FTA at design level and HSI at implementation level) are extended to integrate security considerations into the development lifecycle. Furthermore, an enhancement for the safety assurance case is proposed to encompass other aspects such as security. Two principles are applied for all these enhancements: (a) modify the methods as much as necessary but as little as possible, and (b) propose a framework for the consistent convergence of the engineering disciplines toward a common development lifecycle for dependable cyber-physical systems.

Publication (2018_Elsevier): *Combined Automotive Safety and Security Pattern Engineering Approach (in press) [98]*, Chapter in Journal Reliability Engineering & System Safety. Elsevier 2018
Content: Automotive systems will exhibit increased levels of automation as well as ever tighter integration with other vehicles, traffic infrastructure, and cloud services. From safety perspective, this can be perceived as boon or bane - it greatly increases complexity and uncertainty, but at the same time opens up new opportunities for realizing innovative safety functions. Moreover, cybersecurity becomes important as additional concern because attacks are now much more likely and severe. However, there is a lack of experience with security concerns in context of safety engineering in general and in automotive safety departments in particular. To address this problem, we propose a systematic pattern-based approach that interlinks safety and security patterns and provides guidance with respect to selection and combination of both types of patterns in context of system engineering. A combined safety and security pattern engineering work flow is proposed to provide systematic guidance to support non-expert engineers based on best practices. The application of the approach is shown and demonstrated by an automotive case study and different use case scenarios.

6.2 Full Text of all Publications

In the following a collection of all relevant papers are completing the thesis.

Holistic safety considerations for automotive battery systems

Helmut Martin, Andrea Leitner, Bernhard Winkler

Abstract

The objective of system safety engineering is to develop a system with no unreasonable risk. To this end, risks caused by the electrical and/or electronic (E/E) system that could potentially harm persons must be analyzed, and appropriate risk reduction measures have to be considered in an early phase of development. This requires a close collaboration between different engineering disciplines in order to specify a comprehensive description of risk reduction and mitigation measures - the safety concept. The international functional safety standard ISO 26262 has to be considered for the development of E/E systems within road vehicles up to 3.5 tons.

This standard focuses on E/E measures and considers other non-E/E measures only after the specification of the safety concept. In contrast, this chapter proposes a workflow for the elaboration of an integrated safety concept including safety measures from different engineering disciplines. Two main lessons learned were that the consideration of all kinds of risk reduction measures in the concept phase improves the understanding of the safety of the overall system, and involving various fields of expertise enables the development of a clear safety concept. This approach will improve the development of the overall system, while complying with the requirements of ISO 26262 for the development of E/E systems. The applicability of the introduced approach is demonstrated on an automotive battery case study, where the influence of various safety measures on the ASIL determination has been taken into account in order to reduce the cost of E/E system development.

Helmut Martin
Virtual Vehicle Research Center Graz, Austria, e-mail: helmut.martin@v2c2.at

Andrea Leitner
Virtual Vehicle Research Center Graz, Austria, e-mail: andrea.leitner@v2c2.at

Bernhard Winkler
Virtual Vehicle Research Center Graz, Austria, e-mail: bernhard.winkler@v2c2.at

1 Motivation

High voltage (HV) battery systems are a central part of battery-powered Electric Vehicles (EVs) or Hybrid Electric Vehicles (HEVs), which are becoming more and more important. One reason is the high energy efficiency of E/E systems and the zero (local) environmental pollution of EVs. Their main disadvantage is the relatively short operation range, which is far less competitive compared to conventional vehicles with internal combustion engines. Conventional vehicles provide good performance and long operating ranges by utilizing the high energy-density advantages of petroleum fuels. HEVs combine the advantages of both technologies. Some of the main targets for batteries to be used in HEVs are low costs, high power density (e.g. 1,200W/kg), very high cycle life time (e.g. 200,000 cycles of charge/discharge), long life time (e.g. 9 years), and safety. With the growing importance of e-mobility, automotive battery systems are becoming more important as well. High power (e.g HEV up to 250kW to provide more dynamic driving torques) and high energy application (e.g. EVs such as Nissan Leaf 36kWh to allow longer driving distances) are already being applied in series-production vehicles. Increasing power and energy while decreasing the battery geometries leads to an increase of potential critical effects in the case of malfunctions.

This paper focuses on safety aspects in the context of safety-critical automotive batteries for EVs or HEVs. Regarding functional safety (safety of the E/E system), the IEC 61508[1][1] is the basic international functional safety standard applicable to all industries. The ISO 26262[2] is an adaptation of this standard that is applicable to the development of safety-related electrical and/or electronic (E/E) systems in the automotive domain. One important aspect of functional safety is the risk of electronic malfunction, e.g malfunction of the battery control unit caused by incorrect inputs or software errors. These malfunctions could lead to hazardous events for passengers, other traffic participants, and uninvolved parties (e.g. fire due to overcharge). The risk of malfunctions has to be lowered to an insignificant risk potential by gaining a clear understanding of possible faults, as well as their causes and effects, and by providing solutions for fault mitigation.

In particular, e-mobility is highly interdisciplinary, whereby risk reduction also results from different technical disciplines (e.g. mechanics, chemistry). This means that system safety has to consist of different safety disciplines as well (i.e. functional, electrical, mechanical, and chemical safety). One example for electrical safety could be the prevention of hazardous voltage through the use of galvanic disconnections or isolation. Mechanical safety aims to prevent the deformation of the battery in the case of an accident through the use of cell housings or the installation location for example. Chemical safety can

[1] IEC61058 - Functional safety of electrical/electronic/ programmable electronic safety-related systems

prevent explosions or fire by using a mechanical venting outlet for toxic gases. All of these measures are applicable for the development of a safe system.

Functional safety covers one vital part of system safety engineering, but it is important to realize that other safety measures have to be considered as well. This paper discusses some of the main issues regarding the safety of HV automotive battery systems on different levels of abstraction such as battery cell, battery module and battery pack. Furthermore, the different development phases from specification and design through production and placement in the vehicle are covered as well.

This chapter is structured as follows: Section 2.1 starts with an introduction to the safety lifecycle following ISO 26262. Section 2.2 describes the technical background consisting of the basic architecture of a battery system, together with potential risks and risk mitigation on different levels of abstraction. To get a better understanding, these safety measures are classified in Section 3. Section 4 introduces a modified workflow, which is used to reduce the required Automotive Safety Integrity Level (ASIL) and thereby also the development costs of the electronic system through the definition of non-E/E measures. Section 5 concludes the work and provides an outlook on how the presented work will be continued.

2 Technical background

This section introduces the topic of functional safety in the context of automotive systems. Furthermore, an overview of an HV battery system architecture is provided, including several basic safety measures form different engineering disciplines.

2.1 Introduction to Functional Safety following ISO 26262

The ISO 26262 safety lifecycle encompasses the principal safety activities during the concept phase, product development, production, operation, service and decommissioning as illustrated in Figure 1.

Figure 1 shows the safety lifecycle and highlights the concept phase and the relevant parts of the product development (marked with red dashed lines). The concept phase starts with the definition of the system (here called item), followed by a Hazard Analysis and Risk Assessment (HA&RA), in which all identified hazardous events are evaluated according to ISO 26262 specific risk assessment criteria (i.e., severity, exposure , and controllability). Current hazard analysis techniques can be classified on a hierarchical structure of a system in bottom-up (e.g. FMEA) and top-down approaches (e.g. FTA).

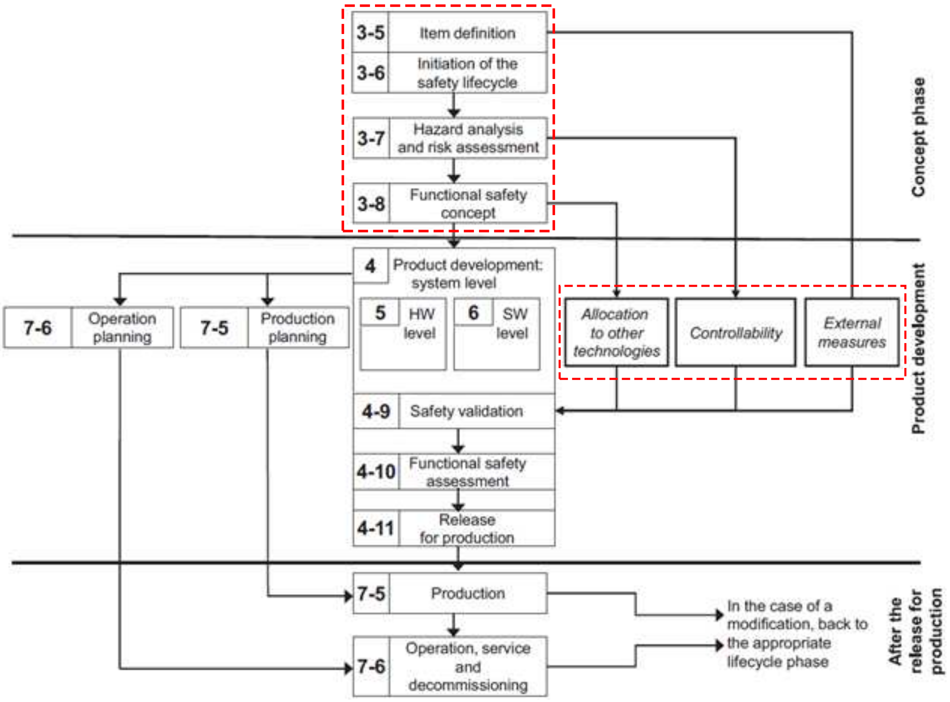

3-5 Item definition
3-6 Initiation of the safety lifecycle
3-7 Hazard analysis and risk assessment
3-8 Functional safety concept
Concept phase
4 Product development: system level
5 HW level
6 SW level
7-6 Operation planning
7-5 Production planning
Allocation to other technologies
Controllability
External measures
Product development
4-9 Safety validation
4-10 Functional safety assessment
4-11 Release for production
7-5 Production
7-6 Operation, service and decommissioning
In the case of a modification, back to the appropriate lifecycle phase
After the release for production

Fig. 1 Safety lifecycle according ISO 26262

The most important, often-cited techniques for performing a hazard analysis are Preliminary Hazard Analysis [3], [4], Concept Failure Mode and Effects Analysis (Con-FMEA) [5], and Hazard and Operability study (HAZOP) [6]. By performing the hazard analysis we identified the following hazards of the battery system: fire/explosion, toxic gases, hazardous voltage of the battery module/pack (U>60VDC), leakage/venting of battery cells (corrosive/toxic (e.g. hydrofluoric acid)), fire (e.g. flammable materials) and, explosion (e.g. breakdown of cell safety vent).

The result of the risk assessment determines the Automotive Safety Integrity Level (ASIL), which indicates the risk of occurrence of a specific failure mode[2] and its necessary degree of avoidance. ASIL values range from ASIL A (low criticality) to ASIL D (high criticality)[3]. Depending on the derived ASIL, the ISO 26262 recommends methods for fulfilling the requirements - higher ASIL leads to higher efforts and costs during the product development.

Based on the results of the HA&RA, safety goals[4] are defined for each hazardous event, and the corresponding ASIL is allocated to each of them. The final activity of the concept phase is the elaboration of the Functional

[2] *"failure mode = manner in which an element or an item fails"* [2]

[3] The class QM (quality management) denotes no requirement to comply with ISO 26262.

[4] Safety goals represent top level safety requirements.

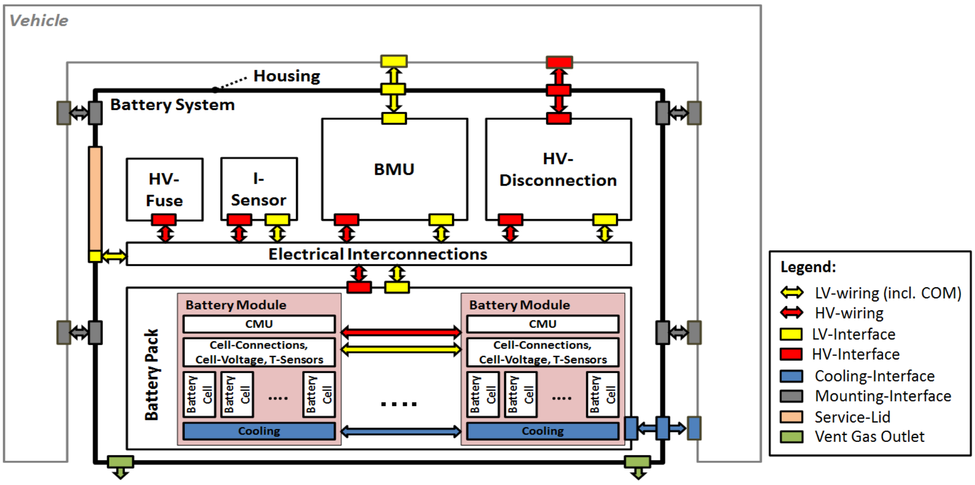


Fig. 2 Battery system architecture - Illustration of the main parts of an automotive battery and their interrelations

Safety Concept (FSC), which defines safety measures that must be fulfilled by the design and development of the system to avoid an unreasonable residual risk. Safety measures are activities or technical solutions used to avoid, control or mitigate the harmful effects of systematic failures and random hardware failures. These technical solutions are implemented by (i) E/E measures (e.g. E/E system with sensor → controller → actuator), (ii) external measures (e.g. organizational measures to counter technical flaws) or (iii) other technologies (solutions from other technical domains, e.g. mechanical fault-back solution), which detect faults or control failure modes in order to achieve or maintain a safe state[5].

2.2 Description of automotive battery system architecture

Figure 2 shows a schematic representation of a system architecture of an HV Li-Ion battery. It consists of the following main components, which already include or represent basic safety measures:

- **Battery Management Unit (BMU):** The main functions of the BMU are the electrical and thermal management, diagnosis functions, insulation monitoring, and the communication with other parts of the vehicle. Electrical management includes charge balancing, charge determination, and the provision of status information, such as system voltage, system

[5] *"safe state = operating mode of an item without an unreasonable level of risk of the system"* [2]

current, or power-time prediction (charging/discharging) for vehicle control functions. Thermal management functionality is used to monitor and evaluate the temperature in the battery system. Disconnection monitoring, charge monitoring, and fault recording represent different diagnosis functions. The insulation monitoring in the battery system is a coordinated function between the battery system and the vehicle.

- **HV Disconnection:** Its main purpose is the disconnection of the battery system from the vehicle HV circuit, and it provides a galvanic separation of the battery and the vehicle in case of deactivation, accident or a safety-critical malfunction. The HV disconnection consists of special HV contactors for the plus and minus terminal. For the activation of the system, a specific pre-charge circuit for both terminals has to be included to realize a soft connection to the vehicle HV circuit. In case of an over-current, an emergency shut-off strategy has to be elaborated because the contactors can only guarantee a limited number of switching cycles under load over their expected lifetime.
- **HV Fuse:** In the case of an over-current, the HV Fuse will disconnect the battery system from the vehicle's HV circuit. Since an over-current causes the HV Fuse to be heated strongly, it must be thermally decoupled from other components (in particular the cells) to prevent a thermal breakdown.
- **I-Sensor:** The I-Sensor provides the current measurement of the whole vehicle HV circuit. The measured current value is used as an input for state-of-charge determination in the BMU and for the thermal management of the battery cells. Each battery has a specific current operation range for charge and discharge. The correct current is measured within this operating range of the battery system with a specified accuracy. If the current is lower or higher than the operating range, a special disconnection strategy has to be implemented with interaction of the HV Disconnection and the HV Fuse.
- **Electrical Interconnections:** This includes all kinds of LV (including the communication) and HV connections between the battery cell pack and the relevant E/E components of the battery system.
- **Battery Cell Pack:** The battery cell pack consists of serial and/or parallel-connected battery cell modules and the battery cell module interconnection.

  - **Battery Cell Modules** consists of battery cells that are connected in series and/or parallel and a cell management unit(CMU). The CMU is responsible for cell charge balancing, measurement of cell voltage and temperature, and the communication between CMUs in different battery modules as well as between CMU and BMU. The cell modules contain a number of redundant temperature sensors to detect areas with critical temperatures. These sensors are connected with the thermal management in order to prevent critical temperature in the battery system.

  - **Battery Cell Module Interconnection** includes all electrical, mechanical, and thermal connections between battery modules.

- **Housing and external interfaces:** The main purpose of the battery housing is to protect the battery system from environmental influences and to protect the driver from any unintended reaction of the battery system. It prevents people from coming into contact with any hazardous voltage. Furthermore, the housing couples the battery system and the vehicle. It has to provide a LV (including communication), an HV interface and an interface for cooling. The housing should provide vent gas outlets (vent gas management), in case of an overpressure in the battery system. For maintenance and repair of the battery system, a service outlet is available. The mechanical mounting interface connects the battery with the vehicle bodyworks.

3 Classification and application of safety measures for automotive battery systems

As mentioned before, it is reasonable to consider different types of measures in order to achieve a more holistic safety concept. Some of these measures are given by customer requirements, while others have to be introduced for additional safety reasons. In this section, we classify them in organizational and technical measures and show some examples.

3.1 Organizational and technical safety measures

This work classifies safety measures in two main categories:

- **Organizational safety measures** [**ORGA**] encompass:

  Safety-compliant development process: The company-specific development process has to cover relevant safety-standard-specific process activities (e.g process audits by external bodies).

  Review/Inspection/Confirmation: Work products that make up the safety case have to be checked by independent[6] parties.

  User safety manuals: Clear and understandable manuals and instructions for the correct handling of the product in the native language of the end user are required.

[6] The degree of independence depends on the safety integrity level, which is defined in the concept phase.

Warning labels and signs indicate potentially critical parts of the system that could cause harm to people (e.g. vent gas outlet at battery housing).

Training: End users have to be informed/trained how to handle the product (e.g. correct driver reaction in the case of malfunction of the battery system). Some kind of safety training is also necessary for first responders in the case of an accident because they should be able to rescue people and should not endanger themselves.

Transport/Storage Regulations: Test and criteria are defined for transport and storage-specific scenarios that have to be approved for the battery cells (UN/ADR regulations e.g.UN 38.3 [7]).

Periodicity of maintenance: The proper functioning of the different safety measures has to be guaranteed until the product's decommissioning. Instructions for maintenance, repair and decommissioning of the product are defined in the standards as well.

- **Technical safety measures**

Functional safety `[FUSA]`: Possible malfunctions of the battery system should be avoided, mitigated, or handled by adequate E/E safety measures (e.g. detection of overcharge of battery and disconnect the battery from any external energy source). This kind of safety measure is explicitly covered by ISO 26262. In contrast, the following other technical safety measures are referred to as external measures or other technologies.

Chemical `[CHEM]`: Any kind of reduction of toxicity of chemical substances (e.g. chemical proof material, cell chemistry).

Thermal `[THER]`: Reduction of thermal energy (e.g. cooling of cells).

Electrical `[ELEC]`: Avoidance of hazardous voltages for customers (e.g. electrical insulation).

Mechanical `[MECH]`: Mechanical construction should prevent or mitigate harm caused by external source.

3.2 Application of measures at battery system units

Not only the incorporation of different engineering disciplines, but also the investigation and coverage of safety at the appropriate level of detail is important for a safe system (see Figure 2).

This section discusses the different levels of units of an automotive battery system. The investigation starts from the lowest level (i.e. the cell) and ends with the highest level (i.e. the vehicle where the battery should be integrated). The battery system is separated into different units, and examples of safety measures are provided.

Level 4 - Battery Cells (BatCel)

This level focuses on all relevant aspects of cell design and structure, cell housing, possible vent gas outlets, cell behavior during aging over life cycle of the battery, and so on.
Sample safety measures:

- `[ORGA]` Cell production process - Establishment of battery cell production quality process, to avoid any kind of contamination of the cell during the production process.
- `[CHEM]` Cell structure - Choice of chemical cell components (e.g. cathode, electrolyte additives).
- `[MECH]` Charge Interruption Device - Mechanical construction in the cell. It is activated if anything causes internal cell pressure to exceed the activation limit physically, and it will irreversibly disconnect the cell from the circuit.
- `[MECH]+[THERM]` Thermal management - Cooling and heating of cells, if needed.
- `[ORGA]+[MECH]+[CHEM]` Vent gas management - Each battery cell provides a defined mechanical venting opening in case of a cell defect.

Level 3 - Battery Module (BatMod)

The battery module level covers various safety measures for the different interfaces of the cells to build up a so-called battery cell stack. One argument for the packaging of cells in modules is the fact that modules can be replaced during maintenance.

Sample safety measures:

- `[MECH]+[THERM]` Use of materials that absorb thermal energy in the module (increase of thermal capacity).
- `[MECH]+[THERM]` Thermal management - Cooling and heating of the cells if needed
- `[ORGA]+[MECH]+[CHEM]` Vent gas management (see level 2).
- `[FUSA]` Monitoring of cell balancing - If a fault is detected (e.g. overcharge), transition to safe state in that situation.

Level 2 - Battery Pack (BatPack)

The battery pack encompasses all modules and provides electrical, thermal, and mechanical connections between them.
Sample safety measures:

- [MECH]+[THERM] Thermal management - Cooling and heating of the cells if needed.
- [ORGA]+[MECH]+[CHEM] Vent gas management - BatPack combines all vent gas channels from each BatMod and leads it to the BatSys.

Level 1 - Battery System (BatSys)

The battery system contains the battery pack, the housing, the battery management unit (BMU), and other relevant components. The BMU internally coordinates all parts of the battery and provides an interface to the E/E system at the vehicle level. It is therefore responsible for the detection and mitigation of errors from the external system.

Sample safety measures:

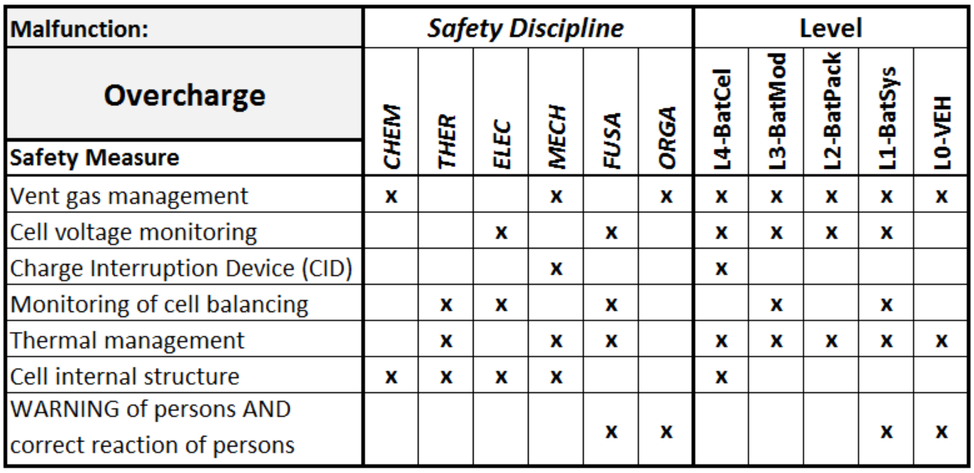
- [ORGA]+[MECH]+[CHEM]+[THERM] Fire extinguisher inlet - The BatSys system should provide an inlet so that the fire brigade could keep the fire at bay and cool down the battery cells.
- [ORGA]+[MECH]+[CHEM] Vent gas management - BatSys provides a vent gas outlet at the battery housing for the vehicle.
- [FUSA] The BMU is an E/E system and is responsible for e.g. monitoring of cell breakdown - If a cell break down is detected by the BMU, several actions should be triggered: disconnection of battery, increase of cooling, communication of critical battery fault.

Level 0 - Vehicle level (target integration of battery system)

At the vehicle level, the prerequisites for the correct functioning of the battery system must be clearly defined. Battery system vendors have to make assumptions about the context in which the battery will be used. These assumptions have to be documented and considered for use. Appropriate safety measures have to be applied in the vehicle, in order to prevent malfunctions in the battery.

Sample safety measures:

- [ORGA]+[MECH]+[CHEM]+[ELEC] Fire extinguisher inlet - The fire extinguisher inlet of the battery system has to be reachable for the fire brigade.
- [MECH]+[THERM] Thermal management - Cooling and heating of the cells, as requested by the battery system.
- [ORGA]+[MECH]+[CHEM] Vent gas management - The vehicle must contain adequate outlet for the vent gas in case of a cell defect.
- [FUSA] Operational Strategy - The vehicle should manage the driving strategy of the powertrain, and critical situations should be prevented by an overall vehicle safety concept (e.g. overcharge, over-temperature).



Malfunction:	***Safety Discipline***						**Level**				
Overcharge											
Safety Measure	***CHEM***	***THER***	***ELEC***	***MECH***	***FUSA***	***ORGA***	**L4-BatCel**	**L3-BatMod**	**L2-BatPack**	**L1-BatSys**	**L0-VEH**
Vent gas management	x			x		x	x	x	x	x	x
Cell voltage monitoring			x		x		x	x	x	x	
Charge Interruption Device (CID)				x			x				
Monitoring of cell balancing		x	x		x			x		x	
Thermal management		x		x	x		x	x	x	x	x
Cell internal structure	x	x	x	x			x				
WARNING of persons AND correct reaction of persons					x	x				x	x

Fig. 3 Example for malfunction *Overcharge*: Mapping of safety measure to battery level and safety disciplines

- `[FUSA]` Warning concept - People in and around the car should be warned by visual and acoustic signals.

Figure 3 shows the sample malfunction *Overcharge* and possible safety measures. It also shows on which entity of the battery systems measures could be applied and the discipline of the measure.

4 Considering non-E/E measures in the concept phase

So far, we have seen that functional safety is just one aspect that has to be considered for the development of a safe automotive system. In this section, we describe a modified version of the ISO 26262 safety workflow, which consists of 3 main activities. Below, these activities and our proposed modifications are described in more detail using the example of an HV lithium-ion battery. This work was conducted in an internal project, and the workflow has previously been published in SEAA2013 [8].

The main purpose of the modified workflow is the holistic investigation of safety measures from different disciplines at an early stage of development. This means that non-E/E measures are already considered in the concept phase, whereas the original workflow sees them as an add-on in later phases.

Basically, three main activities are considered here, as illustrated in Figure 4: (1) Item definition, (2) Hazard Analysis and Risk Assessment, and (3) the design of the Functional Safety Concept. Below, these activities and the newly introduced iteration loop are described in more detail.

1. **Item (system) definition**, the first activity in the concept phase, starts with the definition of the item - the system, its functions on vehicle level,

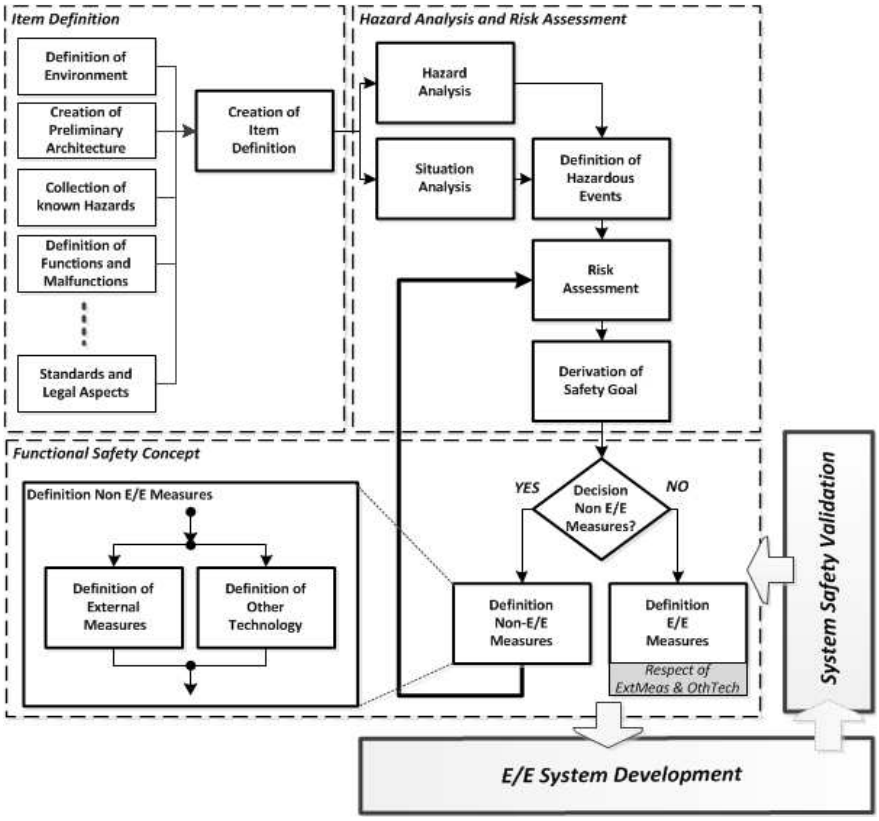

Item Definition
Definition of Environment
Creation of Preliminary Architecture
Collection of known Hazards
Definition of Functions and Malfunctions
Standards and Legal Aspects
Creation of Item Definition
Hazard Analysis and Risk Assessment
Hazard Analysis
Situation Analysis
Definition of Hazardous Events
Risk Assessment
Derivation of Safety Goal
Functional Safety Concept
Definition Non E/E Measures
Definition of External Measures
Definition of Other Technology
YES
Decision Non E/E Measures?
NO
Definition Non-E/E Measures
Definition E/E Measures
Respect of ExtMeas & OthTech
System Safety Validation
E/E System Development

Fig. 4 Workflow of the concept phase following ISO 26262, including item definition, hazard analysis and risk assessment, and the functional safety concept. *Note: A proof of the controllability parameter, which is needed during the safety validation, is not illustrated in this figure.*

and its boundaries to other items. The item in this example is an HV lithium-ion battery. The battery should be used in a Plug-in Hybrid Electric Vehicle (PHEV) with an installed capacity of 24Ah. Potential risks of the Li-Ion battery are hazardous voltage (U>60VDC), leakage/venting (corrosive/toxic, flammable, explosive), fire, and explosion.

First, all relevant and available data concerning the item (e.g. previous projects, customer requirements, state of the art, market analysis, etc.) need to be collected and analyzed. The *Li-Ion Batteries Hazard and Use Assessment Report* [9] provides a very good overview of possible hazards, failure modes and hazard assessment, applicable standards for the US market, and fire protection strategies.

It is further necessary to specify non-functional requirements with regard to standards and legal aspects. In our basic project, we scrutinized several standards (e.g. ISO 26262 for automotive electric/electronic systems and the ECE R100[7] for battery electric vehicles). Based on the results, we created a preliminary architecture to get a better understanding of the interactions between the various parts and to identify functions and malfunctions. Known hazards from other projects and previous experiences have been considered to verify and complete the description. All the results of this step are a fundamental input for the following safety activities.

2. The **Hazard Analysis and Risk Assessment** starts with the analysis of situations and possible hazards, as identified in a preliminary hazard analysis. The following situation analysis aims to identify all driving situations, and the combination with possible hazards leads to hazardous events. Driving situations contain all reasonable combinations of operational, environmental, and weather conditions. The hazard analysis targets the identification of potential hazards for the item on the top level of the system.
We used a Con-FMEA, a systematic method recommended by ISO 26262, to identify the potential hazards of the HV battery system. This approach provides support for traceability, the possibility to verify the completeness of the hazard analysis, and the extension of the Con-FMEA for other FMEAs in the following development phases, as shown in Figure 5. This means that the causes of the failure modes of the Con-FMEA form the new failure modes for the System FMEA. The connections between the identified hazards and the different kinds of failures at different levels of development builds up a complete failure net. This failure net is a step-by-step refinement in the FMEA, which supports failure propagation and traceability[8].
In our example, the hazard and situation analysis resulted in 640 hazardous events. These hazardous events were identified by a stepwise combination and filtering of possible combinations of operational, environmental, and weather conditions. Finally, the plausibility of each combination was checked. As a result, we identified 121 plausible hazardous events, which were then assessed according to the risk assessment parameters *Severity (S)* [S0..S4], *Exposure (E)*[E0..E4], and *Controllability (C)*[C0..C3]. If any of these parameters results in a "S=0 OR E=0 OR C=0" no safety development is needed - the level QM (quality management) is sufficient. The rationale behind each classification has to be documented appropriately because it is the basis for the ASIL determination, according to the risk graph of ISO 26262 (see Figure 6).
Finally, safety goals have to be specified depending on the hazardous events and risk assessment results.

[7] ECE R100 - Uniform provisions concerning the approval of vehicles with regard to specific requirements for the electric power train

[8] For the creation of the FMEAs we used the tool APIS IQ-RM PRO FMEA v6; http://www.apis.de

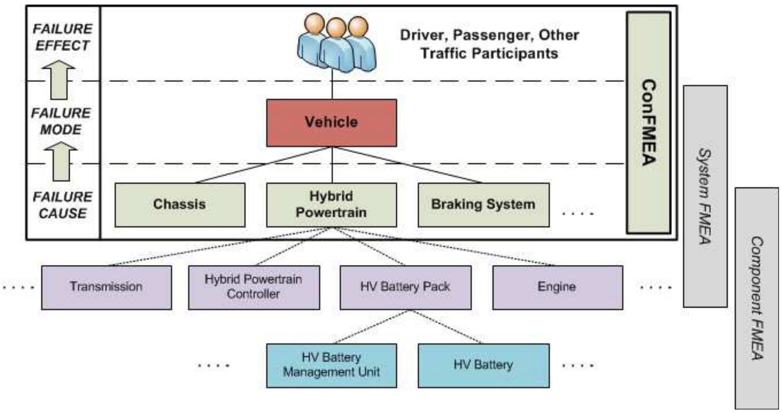
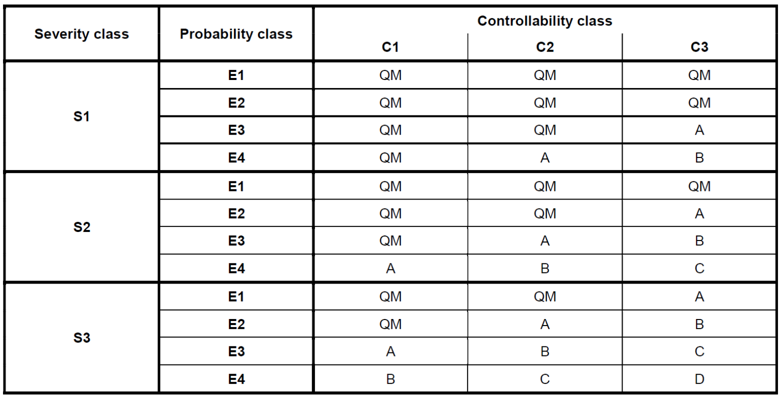


Fig. 5 FMEAs applied on different development levels



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

Fig. 6 Risk graph for ASIL determination according to ISO 26262[Part3]

Below, an exemplary classification of a hazardous event is shown, where the vehicle is deactivated:

- **Hazardous event:***Fire because of internal cell defect during parking situation (system is deactivated)*
- **Individuals at risk:** *Persons around the vehicle (Assumption: There is noone in the vehicle during the parking situation.)*
- **Possible harm:** *Burning of cell could cause hot smoke gas that could lead to smoke gas contamination and burns of critical injury degree are possible.*
- **Perception:** *Unpleasant sweet smell, and visible smoke*
- **Severity:** ***S2*** - *Severe injuries possible (life-threatening, survival probable)*
- **Exposure:** ***E4*** - *The vehicle will park every day for a long time in the parking garage.*
- **Controllability:** ***C3*** - *Less than 90% of all drivers or other traffic participants are usually able, or barely able, to avoid harm.*

One main challenge here is the fact that the E/E system of the PHEV is deactivated during parking. For this specific situation, it is not possible to fulfill the safety goals with E/E measures only because these measures mainly mitigate hazardous situations during operational modes.
We derived the required ASIL for our exemplary hazardous event using the risk graph (Figure 6)of ISO 26262 : *Severity S2*, *Exposure E4* and *Controllability* of *C3* → *ASIL C.*
The last step is the derivation of safety goals, as in this case *"'Avoidance or/and mitigation of hazards caused by internal cell defect.*" with the safe state *"No fire outside of the vehicle.*"

3. The **Functional Safety Concept (FSC)** describes the derived safety measures (see Figure 7) which realize the safety goals. Following ISO 26262, there are three different types of safety measures (E/E safety measures, other technologies and external measures). One viable approach to fulfill the safety goals in this case is the consideration of non-E/E measures in order to reduce the required ASIL.
Our modified workflow introduces an additional decision regarding whether or not it is possible or better to define non-E/E measures to fulfill the safety goals. If it is, we propose the identification of non-E/E safety measures with support from specialists from other disciplines (e.g. mechanical engineering). They need to be involved at an early stage of development because, based on their expertise, external measures and other technologies can be elaborated and considered. An example of another technology measure for an HV battery is the use of *fire-resistant materials for the battery housing* and an external measure could be a *fire detector in the parking garage.* All kinds of safety measures have to be introduced in the FSC as Functional Safety Requirements, which are linked to the corresponding elements of the Functional Safety Architecture. The main elements of the identified E/E measure are a sensor, a processor and an actuator. The FSC should provide a safety event chain from the detection of critical signals (sensor) to the processing and correct decision for the safe operation (processor), and finally the execution of a safe state (actuator) defined in the top level safety goal for the specific hazardous event.

4. **Iterative refinement step, including update of functional safety concept** - After applying the different safety measures, we introduce a feedback step to repeat the risk assessment with the new conditions. The following measures were defined for the HV battery example: (1) External measures: *Fire detection unit and fire extinguisher have to be installed in the parking garage*, and (2) other technologies: *Fire-resistant housing of the battery system.*
The introduction of these measures changes the risk assessment as follows:
   - **Severity**: ***S1*** - *Light or moderate injuries possible (not life-threatening)*

- **Rationale for new S**: *The use of special fire-resistant materials for the mechanical construction of the housing will reduce the intensity of the harm.*
- **Controllability:** ***C2*** *- 90 % or more of all drivers or other traffic participants are usually able to avoid harm.*
- **Rationale for new C**:*People will be warned by acoustic signals from the fire detection unit; a fire extinguisher will be available to extinguish the fire; the fire brigade will be alerted in the case of fire.*

This leads to the new rating result of an ASIL A classification for the hazardous event. Lowering the required ASIL from *C* to *A* means that the remaining risk which has to be covered by E/E safety measures is lower, and therefore a less complex E/E measures and less development effort are needed.

The last step is an update of the functional safety concept. Each introduced safety measure that contributes to the risk reduction is specified as functional safety requirements, which are mapped to the elements of the functional safety architecture. See Figure 7 for the main parts of the functional safety concept.

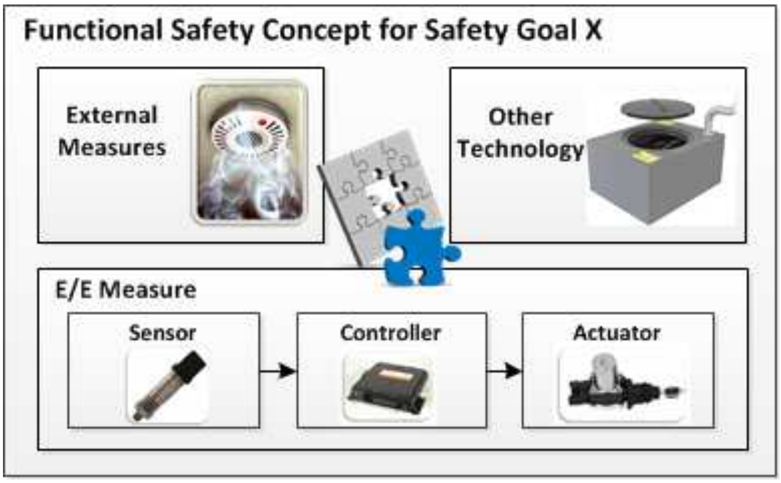


Fig. 7 Principle of Functional Safety Architecture consisting of three types of safety measures

5 Discussion and Conclusion

This paper summarized our investigations of functional safety based on ISO 26262 for HV batteries typically used in EVs or HEVs. We presented an approach for an iterative determination of the required ASIL by applying non-E/E measures. We observed that it is often productive to consider external measures and other technologies early in the concept phase, and that the incorporation of different engineering disciplines with different viewpoints helps to improve the safety of the entire system.

Functional safety ≠ system safety
One main observation of this work was that hazards and risks result from different technical disciplines because e-mobility is highly interdisciplinary. Functional safety covers one part of this overall system safety. We identified several other types of safety that are relevant in this context, i.e. electrical safety (e.g. considering hazardous voltage), mechanical safety (e.g. concerning the deformation of the battery in the case of an accident) and chemical safety (e.g. helps to prevent explosion or fire). One main finding of this project is the importance of a strong interaction of all these different safety disciplines in the concept phase, which requires an organizational safety culture that fosters interaction between different disciplines. Not all hazardous events can be covered by E/E safety measures alone. Other technologies or external measures are equally important in order to achieve a safe system state.

Intercultural aspects
The discussion with other departments results in a more holistic, interdisciplinary system and safety understanding. It also reveals how each team is able to contribute to the safety of a system. A discussion at an early stage of the project definitely improves the interaction between the different teams. Nevertheless, it has to be kept in mind that different views include different opinions, and often even contradicting opinions. All of them are correct in their specific systems or safety views. This can result in never-ending discussions, if there is no clear moderation.
We can offer one example of a discussion about the definition of the safe state of the system. One common function of the battery is *charging*. In the case of *overcharge*, the engineers responsible for electrical safety define the protective safe state in any such case for the electrical system to disconnect the HV battery from the high voltage net of the vehicle. This would lead to an undefined operation condition of the vehicle. The functional safety team must think about any possible driving situations, where an unintended loss of high voltage energy could lead to a critical situation. One such situation could be an overtaking maneuver on a country road, where a significant loss of driving torque could lead to a dangerous situation for the driver or other traffic participants.

Scope of functional safety
With a holistic safety view, it is often difficult to define the responsibilities for different hazards. Sometimes hazards are not directly caused by an E/E failure, but are an indirect consequence of a malfunctioning E/E system. Regarding the example of electric or hybrid electric vehicles, it cannot be clearly

defined whether or not the HV battery should be considered only as an E/E system.

Acknowledgment

The authors would like to acknowledge the financial support of the "COMET K2 - Competence Centres for Excellent Technologies Programme" of the Austrian Federal Ministry for Transport, Innovation and Technology (BMVIT), the Austrian Federal Ministry of Economy, Family and Youth (BMWFJ), the Austrian Research Promotion Agency (FFG), the Province of Styria and the Styrian Business Promotion Agency (SFG). The research leading to these results has received funding from the ARTEMIS Joint Undertaking under grant agreements pSafeCer n∘ 269265 and nSafeCer n∘ 295373. Further, the authors would like to acknowledge APIS GmbH for their support by their software tool, APIS IQ FMEA PRO.

References

1. *IEC61058 - Functional safety of electrical/electronic/ programmable electronic safety-related systems*, 1st ed. International Electrotechnical Commission, 2005.
2. *ISO26262 - Road vehicles - Functional safety International Standard ISO 26262:2011 (Parts 1-10)*, 1st ed. ISO copyright office, 2011.
3. C. A. Ericson, *Hazard Analysis Techniques for System Safety*, 1st ed. John Wiley Son, 2005.
4. R. Mader, G. Grießnig, A. Leitner, C. Kreiner, Q. Bourrouilh, E. Armengaud, C. Steger, and R. Weiß, "A computer-aided approach to preliminary hazard analysis for automotive embedded systems," in *Engineering of Computer Based Systems (ECBS), 2011 18th IEEE International Conference and Workshops on.* IEEE, 2011, pp. 169–178.
5. *FMEA Handbook Version 4.1*, 1st ed. Ford Motor Company, 2004.
6. N. Leveson, *SAFEWARE system Safety and Computers*, 1st ed. Addison-Wesley Publishing Company, Inc, 1995.
7. *UN Recommendations on the Transport of Dangerous Goods, Manual of Tests and Criteria 38.3 Lithium batteries, Rev.5, Amend.1*, 5th ed. UN Recommendation, 2009.
8. H. Martin, B. Winkler, A. Leitner, A. Thaler, M. Cifrain, and D. Watzenig, "Investigation of the influence of non-e/e safety measures for the asil determination," in *Software Engineering and Advanced Applications (SEAA), 2013 39th EUROMICRO Conference on.* IEEE, 2013, pp. 228–231.
9. C. Mikolajczak, M. Kahn, K. White, and R. T. Long, "Lithium-Ion Batteries Hazard and Use Assessment; Final Report," Exponent Failure Analysis Associates, Inc./ Fire Protection Research Foundation, Tech. Rep., July 2011.

SAE International®

System Modeling for Integration and Test of Safety-Critical Automotive Embedded Systems

2013-01-0189
Published
04/08/2013

Martin Krammer, Helmut Martin, Michael Karner, Daniel Watzenig and Anton Fuchs
Virtual Vehicle Research and Test Center

doi:10.4271/2013-01-0189

ABSTRACT

Functional safety of automotive embedded systems is a key issue during the development process. To support the industry, the automotive functional safety standard ISO 26262 has been defined. However, there are several limitations when following the approach directly as defined in the standard.

Within this work, we propose an approach for the integration and test of safety-critical systems by using system modeling techniques. The combination of two state-of-the-art modeling languages into a dedicated multi-language development process provides a direct link between all stages of the development process, thus enabling efficient safety verification and validation already during modeling phase. It supports the developer in efficient application of requirements as defined by ISO 26262, hence reducing development time and cost by providing traceable safety argumentation.

Based on a hybrid electric power train scenario, we evaluate the benefits of the proposed system modeling approach for early verification and validation of safety-critical embedded systems.

INTRODUCTION

Automotive electric and electronic (E/E) systems are key drivers for innovation in today's vehicles. While new functions are delivering eco-friendliness (hybrid and pure electric vehicles, etc.), assistance/comfort (drive-by-wire, park-assist, etc.) and active safety (electronic stability control, lane-change-assist, brake-assist, etc.) their inherent complexity is challenging manufacturers and suppliers. At the same time, functional safety of these products is a key issue: During the whole car's product life cycle, there are many potential risks for physical injuries, or even worse, fatalities. Therefore, these potential sources of harm should strictly be avoided.

For these reasons, the latest automotive safety standard ISO26262 [1] is targeted at E/E systems of passenger cars. Structured requirements and recommendations are leading to documents, which are intended to confirm that the E/E system developed is reasonably safe. However, following this approach directly has some limitations.

First, a lack of traceability can be observed in traditional engineering approaches between safety goals, their derived safety requirements, created work-products, developed components, and the resulting system. This complicates the argumentation of a product's safety. Second, the automotive supply chain must be aligned to support these safety activities. Horizontal integration issues (e.g. different software-components) and vertical integration issues (e.g. component, module and system levels) as well as testing problems arise in this context. Integration test cases applied during document-centric approaches might not be as correct and complete as they seem, due to missing links to the initial safety goals.

To overcome these issues, we propose a combined solution covering two main aspects, namely system modeling and virtual prototyping. For a formalized system description approach, we decided to deploy SysML and SystemC languages. These languages complement each other and provide the possibility for model-centric development. They support the need for the characterization of structure, behavior, requirements and simulation of automotive embedded systems.

The resulting models are used as a reference throughout requirements- and system design phases and serve as a basis for component design and implementation phases. Different views on the model are used to extract necessary information for integration and test scenarios, which span across horizontal and vertical levels.

The desired functions are rooted in different engineering domains. Therefore a co-simulation approach is followed: By coupling different simulation tools, the impact and interaction of hardware-, software-, and mechatronic/mechanical components on the vehicle's functions can be observed. That approach provides the possibility to perform fault injection to expose safety goal violations in early development phases. In context of functional safety, the created models and corresponding test scenarios are oriented at the overall safety goals and form an executable safety case, providing arguments for safety validation. The introduced concepts and methods are demonstrated by an automotive use case.

The innovative approach described within this work is an immediate output of the VeTeSS[1] project, involving 24 partners from eight countries, including OEMs, SMEs and research institutions. The VeTeSS consortium works on standardized tools and methods for verification of the robustness of safety-relevant automotive systems. Bringing together partners from every part of the supply chain, VeTeSS develops automated, quantitative processes usable at all stages of development.

RELATED WORK

ISO 26262 [1] is the most recent functional safety standard targeted at electric and electronic systems of series-production passenger cars. It was released in late 2011. Its last part, number 10, followed in mid 2012. It reflects the latest state of the art in automotive engineering.

In this paper, a combined language SysML/SystemC approach is followed, with respect to the needs of the automotive engineering domain. In the fields of integrated circuits and systems-on-chip, similar approaches have shown to be successful [2]. For system-on-chip design the authors of [3] are showing a SysML-to-SystemC transformation procedure, allowing an automated generation of SystemC code. They are confident that their approach is suitable in early system design phases. However, related work in this field is mostly based on low-level hardware and/or software architectural design, without the scope of higher level system design, including the view on a system's functions and safety properties.

SysML was used for safety related analyses in the past. In [4] an FMEA analysis was conducted based on two different diagram types. An interesting approach is also discussed in [5]. A SysML modeling process for software, electronics and mechanics is introduced. Various safety aspects are targeted and a preliminary hazard analysis is conducted. This approach is to be industrialized; Software products like Medini Analyze[2] already provide limited support for SysML-based safety analyses.

The general topic of re-use in scope of ISO 26262 is covered with the introduction of *safety elements out of context* (SEooC). According to its definition, it addresses the development of new components, which are not targeted at a specific car intended for series production. Opposite to this, the process of qualification tries to evaluate the status of existing components for integration. Examples for development and integration practices of SEooC are given in [6].

SysML is a general purpose modeling language. It is based on Unified Modeling Language (UML), and was constructed for systems engineering applications. It is standardized by the Object Management Group (OMG) [7].

SystemC is a hardware description language targeted at digital system design. It fills the gap between typical hardware description languages like VHDL or Verilog and higher level modeling languages. It is filed as IEEE standard 1666 [8].

METHODOLOGY

Language-Tool-Process Strategy for Modeling Automotive Embedded Systems

Various approaches exist for modeling automotive embedded systems and some of them led to recognized industry standards. We focus our research on the development of safety-critical systems and functions. Examples for such are completely electric or hybrid electric power trains, various x-by-wire functions, or comfort functions like an electric park brake or power window lifts. Those are clearly safety-relevant, because of their immediate interaction with the driver or the surrounding environment. Thus, we are striving for a holistic approach, not only involving pure electric and electronic systems and software, but also mechanic or hydraulic systems.

In order to support our diversified approach, we chose to evaluate SysML for deployment in industry projects. In today's industry projects, simulation as primary method for verification plays an important role. In the context of functional safety, we are also eager to see if our final product meets real world expectations and behaves as intended,

[1] http://www.vetess.eu
[2] http://www.ikv.de/

excluding any misbehavior. Therefore, advanced validation methodologies targeted at functional safety need to be applied. Since SysML is not intended for comprehensive simulation purposes, we thought of a supplement in order to meet these goals. We have chosen SystemC to complement SysML, for the reasons explained in the next section.

SysML models are based on diagrams, extended from the UML specification. SysML uses blocks, derived from UML classes, for system architecture definition. System behavior is specified with use case diagrams, activity diagrams and state machines. For a SysML capable tool we decided to use Enterprise Architect[3] as primary modeling tool. It allows the customization of the SysML language specification and the generation of models, both according to the OMG meta-data architecture.

Evaluation of Language Properties

Standardization

SystemC became an IEEE standard in 2005. Version 2.2 was published by the "Open SystemC Initiative" (OSCI) in 2007. The most recent version is 2.3, published by "Accellera Systems Initiative". SystemC addresses discrete-value systems only, with its extension SystemC-AMS [9] it may be used for analog systems as well. As of 2012, SystemC-AMS is available as a 2.0-draft version. SystemC is a C++ library providing methods and data types for the modeling of systems, hardware and software. Minimum requirement is one of a broad variety of compilers. Any C++-compatible development environment is suitable for SystemC. SysML became an Object Management Group (OMG) standard in 2007 with the release of version 1.0. Tool support is provided.

Forward Engineering Possibilities

Because SysML is derived of UML, forward engineering possibilities are given in terms of code generation. SysML itself does not offer simulation features by specification, but some supportive tools have capabilities to transform activity diagrams and state charts to executable code. Constraint evaluation can be considered possible by parametric diagrams. SystemC offers excellent forward engineering possibilities. Architectural models may be filled with abstract behavioral designs, transaction level models, detailed software- or hardware blocks or even a concrete software implementation. Graphical user interfaces are available as well. For simulation, a simulation engine is included next to the models themselves within the executable. Constraint evaluation features are included, either by the compiler, or via external libraries, like the SystemC Verification Library [10]. Simulation speed is another point underlining the efficiency of SystemC. Due to low level code execution, SystemC and its analog extension have shown to be very fast [9]. But still, extensive use of hardware modeling concepts slows simulation speeds down, but on a high performance level.

Formality

SysML can be considered as a semi-formal design language, which allows definitions as clear as the designing engineer wants them to be. Data- and energy flows between blocks are defining interaction. Opposite to this, SystemC is a formal programming language using well defined blocks, interfaces and data types, which provide clearly more uniqueness.

Applicability Hardware/Software/System

SystemC is advantageous concerning implementation levels. It does not matter whether a system comprises software and/or hardware. Systems are built block wise, using a hierarchical design approach with clear interface definitions. Hardware can be described using digital and analog models, on register transfer or behavioral levels. The concept of time is fully implemented, including concurrency for hardware designs. For software, all types of C or C++ code can be adopted by simple inclusion. SysML allows the interdisciplinary description of hardware and/or software systems as well and supports the allocation of requirements. Various aspects, which are implicitly present in SystemC models, are expressed in a more explicit way through specific diagrams. System constraints or state machines for example, are more legible in a dedicated SysML diagram than in SystemC code, although they may express exactly the same. Hardware designs can be drafted by using state charts or flow diagrams. A clear definition of time-related concepts is missing. Software designs are supported in a more native way, since the derivation from UML. However, there is no involvement of code.

Industrial Use

SystemC is primarily utilized in the semiconductor industry, but has gained some interest in other domains. SysML users can be found in various domains. In the automotive industry, both languages are not fully recognized by manufacturers, but suppliers are making use of it.

Summary

All in all, SysML offers a high level of abstraction, very suitable for conceptual designs, overall system design and requirement allocation. Its forward engineering possibilities are considered to be tool-dependent. SystemC on the other hand, is capable of describing clear, unique system designs, with formal interfaces and module behavior. Its simulation capabilities allow model verification.

Figure 1 shows a variant of the V-model, where the core capabilities of both languages can be seen. SysML has

[3] http://www.sparxsystems.com

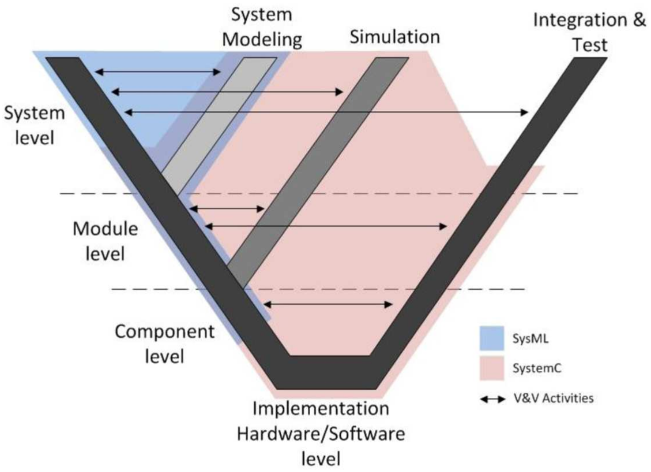


Figure 1. Language capability evaluation

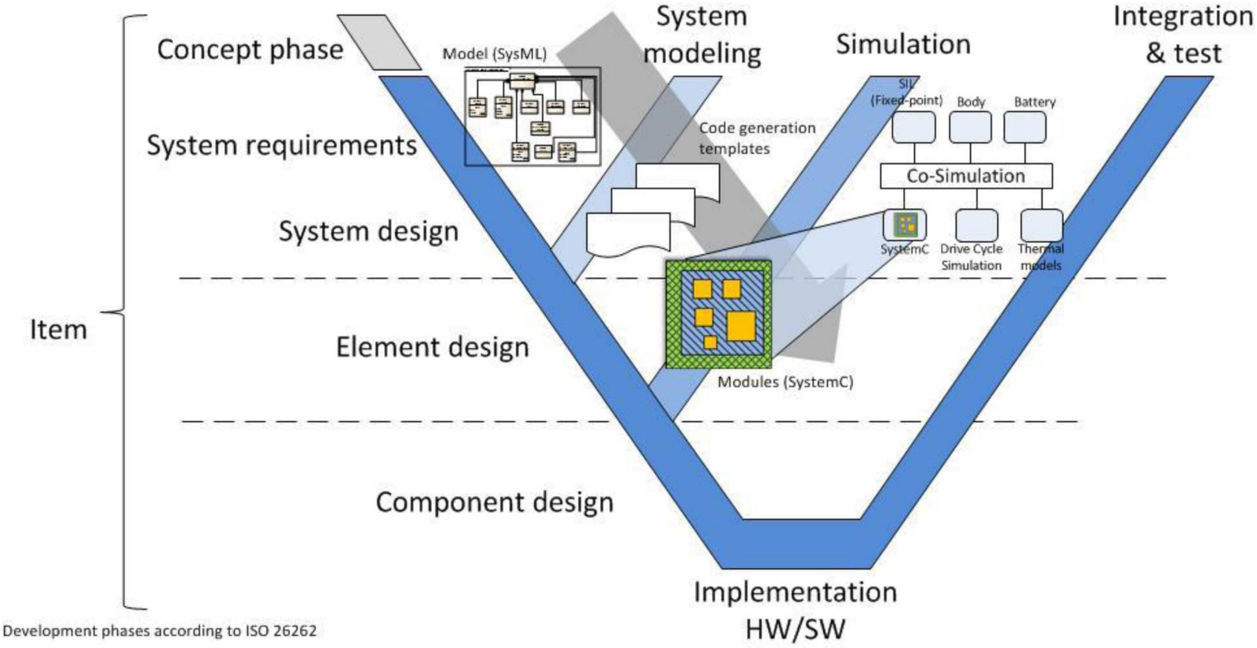


Figure 2. Development process overview

advantages when it comes to the allocation of requirements to system components. There is a strong overlap in system design and modeling. SystemC has its primary strengths in applied element and component design, simulation and verification. A combined approach of both languages seems to be appropriate to cover the entire V-model. Due to the allocation of functional requirements and safety requirements to hierarchical entities and their verification in simulation, this seems an appropriate way to deliver necessary arguments to support a product's safety case.

Multi-Language Development Process

V-models are commonly used in the automotive engineering domain. In order to sum up our findings about SysML and SystemC, a V-model is proposed. It is shown in Figure 2.

On the top left hand side of the V-Model the concept phase is denoted. In terms of the ISO 26262 safety life cycle, it represents a vital phase for the following system development activities. It starts with the definition of the so called item, which is the representation of a system that implements a function on vehicle level (e.g. hybrid control unit). The item definition gathers all the safety related information for characterizing the system (e.g. functions, known hazards …) and sets the boundary to the other vehicle items/systems including all interfaces and possible interactions. In terms of SysML, the use case diagram helps to characterize the item and its behavior. The embedded activity diagrams allow a more detailed specification of system states and state transitions.

Based on that information, the Hazard Analysis and Risk Assessment (HARA) has to be performed. It is used to identify all possible hazards and couple them to drive and other operational situations. Next, these hazardous events are classified by three independent categories: Severity of harm (S), probability of exposure (E) and controllability (C) of these hazardous situations. The quantitative evaluation of these metrics has to be supported by meaningful and verifiable argumentation. As a result of the HARA, the ASIL can be determined for each hazardous event by taking the main parameters S, E and C (ASIL = f(S,E,C)). In that relationship the ASIL represents a determined risk of occurrence of a specific failure mode and the necessary degree of avoidance of that failure mode. The ASIL is classified from A to D, where ASIL A means the lowest and ASIL D the most stringent level. This illustrates the importance of the ASIL determination, because the higher the ASIL, increased development efforts (e.g. required safety analysis techniques, diagnostic measures in system design…) and costs for development and justification of the safety related system arise.

Based on the results of the HARA the safety goals have to be determined for each hazardous event with their corresponding ASIL. These are the top level safety requirements of the item, and can be introduced to SysML as a normal requirement. Additional stereotype definitions are made to clearly separate safety goals from functional safety requirements, technical safety requirements and component requirements. Each requirement includes additional details with relevant attributes, like operational situation, fault tolerant time interval or physical characteristics (e.g. the maximum levels of unwanted behavior). After verification of the HARA and determination of safety goals a functional safety concept is elaborated. During that activity, functional safety requirements are derived from the safety goals and allocated to preliminary architectural elements from the concept phase. This process is also supported by our model. Requirements may be refined, by deriving sub-requirements and by creating relationships between them. In the functional safety concept, all safety measures are specified, including safety mechanism of the item itself, external safety measures or measures of other technologies. So the functional safety concept provides an overview of all identified measures (arguments) that have to be satisfied in the technical realization during the system development. The functional safety concept has to be considered within two activities. First, it is an input for the elaboration of (technical) system requirements during the product development phase on system level. The customer requirements and functional safety requirements have to be satisfied in the technical safety concept. Second, it provides the assessment criteria for the system safety validation activities of the item to get the final evidence that the item is acceptably safe on vehicle level. By assigning the identified requirements to their corresponding architectural entities, activities or system states, an increased level of traceability is given during the entire development process.

On the very left hand side of Figure 2, structural entities of ISO 26262 can be recognized. An item is used as top-level entity, to implement a function on vehicle level. It is composed of systems or entire arrays of systems. Systems can be further divided to elements. Elements in turn, relate at least a sensor, controller and actuator with each other. ISO 26262 allows an element to be a system again, which deepens the resulting hierarchy. Otherwise, elements include components, which are no system level elements. A key property is that they are logically and technically separable.

On component level, software and hardware is differentiated. Software components include at least one software unit, which is considered to be an atomic component of the architecture. Concerning software testing, it is applicable for stand-alone tests. Hardware components include more than one hardware part. In terms of architectural system design, parts cannot be subdivided any further. Examples thereof are simple resistors and also integrated circuits.

The bottom-line implementation level is usually covered by domain specific development steps, out of control systems engineering, digital design, signal processing and the like.

This four-staged meta-model can be applied to generate arbitrary composites of embedded systems. However, what ISO 26262 refers to the term architecture is usually defined by three of them, namely item, system and element. Our model supports this structuring by extending block definitions. This way, entity boundaries can be drawn arbitrary between different engineering domains. The rightmost hand side of the V-model covers tasks of integration and test. While integration of software components or hardware units can be done on supplier level, integration tasks on vehicle level require the availability of a prototype car. For several reasons we do not strive for vehicle level integration. Besides availability, prototypes have shown to be costly. Every stage of development, which can be accomplished using simulation, helps to save time and cost. Thus we spare the rightmost integration and test phases in the V-model and draw a parallel arm covering simulation techniques. Since we focus on system modeling and design, another arm covering these topics is drawn in parallel.

Language & Tool Coverage

In the following, we discuss how a combined SysML and SystemC approach, as depicted in Figure 2, has the potential to support our needs for the efficient design and validation of safety relevant embedded systems.

Concept phase and requirement engineering

Requirements are adopted written in natural language. For analysis and consistency improvement, industry relevant tools like APIS IQ[4] can be applied during this phase. These requirements will then be allocated to their corresponding entities. The system model created in the right branch of this upper left V-model has to reflect these allocations. Furthermore, requirements must be verified in the system design phase. This is a key issue to ensure continuous development and coherent argumentation of the final product's functional safety properties.

System Design

The primary goal during this stage of development is an appropriate architectural design, which usually implements functional and technical requirements. Furthermore, system design is about the breakdown of large systems into smaller subsystems. Concerning structure, the model allows the definition of block definition diagrams (bdd) and internal block diagrams (ibd). Especially internal block diagrams contain information how blocks are connected and which interfaces they share. Code generation templates related to blocks translate architectural decisions into SystemC modules. These are connected with unique signals and channels. The logical link between SysML model blocks and SystemC modules is formalized and can be traced throughout the entire design process.

Element design

In context of ISO 26262, an element includes hardware or software in shape of components.

Component design and implementation

Domain specific tools are used to carry out this phase. Mathworks Matlab, dSPACE TargetLink and others are prominent candidates for various engineering purposes.

Re-Use of Components

Another aspect relevant for the V-model is the re-use of components, which are developed in the scope of ISO 26262. Those “safety elements out of context” are developed prior to the design of a vehicle intended for series-production. The authors of [6] have shown an industrial approach supporting the development and integration of SEooC.

In the hierarchy proposed in Figure 2, the levels affected by the design aspect of re-use are the system-level and the element-level. Based on the management of functional requirements and safety requirements, certain assumptions about the future use of the SEooC are determined, and documented in the product's safety manual. From the integrator's perspective, his/her requirements should meet the given assumptions. Concerning safety aspects, the liable integrator pulls the necessary evidence from the safety manual to construct an argument.

For all these reasons, the interface definition of SEooC plays an important role. During system design, modeling and simulation phases this aspect is of particular interest.

Tool Chain & Co-Simulation Support

The previously explained common V-diagram as a process model does not make any assumptions on tooling, since SysML and SystemC are pure languages. In this section, we propose a workflow, mainly based on the strong connection between SysML, SystemC and co-simulation.

When the idea for a novel automotive product comes up, a set of requirements is initially created. In the beginning, these are manageable without any limitations caused by architectural aspects. Tools like DOORS[5] support these steps. When the aspect of functional safety is added, a subset of requirements is paid special attention to. Most important about this phase is the derivation of automotive safety integrity levels (ASIL) per desired function. Based on preliminary architectural assumptions, requirements and safety integrity levels are assigned to their corresponding architectural entities. To accomplish these steps, we have chosen Enterprise Architect as primary modeling tool for SysML. The various types of requirements can be collected, structured and regrouped within a system model. Its export capability to XMI data format and the application of transformation templates allows the perpetuation of architectural decisions from SysML to SystemC language. By compiling the SystemC code, the Microsoft C++ compiler builds the simulation engine for SystemC. At this stage the preliminary architectural assumptions and system design results turn into concrete executable models for the first time in the proposed V-model.

This point has a second advantage. By receiving an executable C++-model, an interface to co-simulation methodology is achieved at the same time. We support the idea of independent co-simulation, introduced by our in-house developed co-simulation software ICOS [11]. This way, the inclusion of domain specific tools to the V-model becomes possible, even on higher levels of abstraction.

[4] http://www.apis.de

[5] http://www.ibm.com

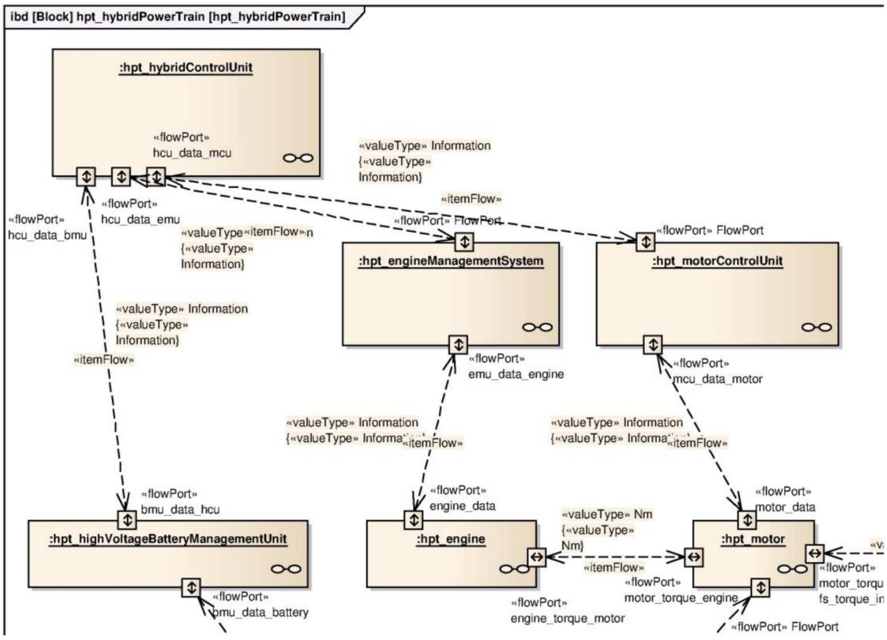


Figure 3. The hybrid control unit (HCU) in context of the system

In the system model, the behavior of components and elements can only be expressed at a very high level of abstraction. The process transition to SystemC code at this stage enables two benefits related to co-simulation. First, it becomes possible to include other relevant models. These models can be existing concrete implementations, thus the idea of component re-use is supported. Second, it becomes possible to replace existing, abstract SystemC models with more complex models. A combination of both possibilities not only allows the evaluation of high-level concepts in a broader environment, but also the traceability of certain features, back through the chain. The fulfillment of safety goals can be evaluated this way. However, an automated validation is tool-dependent, and can be quite different from domain to domain.

Examples for that are vehicle body electronic and electric components, chemical or electric battery models or drive cycle simulations. Fixed-point software-in-the-loop (SIL) approaches may also be realized. Especially for full electric or hybrid electric vehicles these approaches offer sophisticated possibilities.

DEMONSTRATOR

In order to demonstrate our approach, a hybrid electric power train is described. It contains a hybrid control unit, controlling all power train functions. The HCU's software is subject to an implementation. Its overall usage scenarios are based on vehicle level and provide the necessary depth down to component level to evaluate our approach. This hybrid electric power train comprises of many different parts from different engineering domains, and therefore suffices our cross-domain system modeling approach.

The entire hybrid electric power train is safety critical, since every driver input shall be carried out as defined and shall move the vehicle as intended. Due to the specification of this use case, the standard ISO 26262 applies.

We consider a vehicle acceleration scenario, where the turbo-charged internal combustion engine (ICE) is assisted by a battery-powered electric motor. In this scenario, the driver accelerates the car at full throttle. In that case, the ICE is not capable of providing the maximum power from the very beginning (turbo lag), thus a boost function is triggered. For a parallel hybrid power train architecture, the boost function adds the torque of the motor to the torque of the engine in an additive way. A similar function is described in [12]. Besides the boost mode, an overboost mode allows to recharge the vehicle's battery right after the boost mode, when the engine operates in an optimal range and provides a sufficient amount of energy. In this context, we consider three safety goals, related to the hybrid control unit and the vehicle's battery:

- *In boost mode, the battery shall only be discharged.*
- *In overboost-mode, the battery shall be charged.*
- *In coasting mode, the battery shall not be charged.*

These three operating modes are specified within our model's HCU. In terms of ISO 26262, the item hybrid power train contains the HCU as a system, which contains a number of software components. The safety goals and subsequent requirements are assigned to these entities. The vehicle's battery must be protected from any functional misbehavior to avoid safety hazards. These were identified during the concept phase (HARA). A typical example would be fire or explosion due battery overload, or battery cell damage due to deep-discharge.

The model's hybrid control unit consists of state chart diagrams, triggering the different operating modes. The resulting SystemC modules are included within an ICOS co-simulation configuration and linked to an instance of AVL Cruise[6] drive cycle simulation. The driver input and the vehicle's overall behavior, based on the HCU model, can now be validated. The validation procedure as well as the entire process automation is still subject to further research.

SUMMARY AND CONCLUSION

The development of safety-critical embedded systems according to standards like ISO 26262 is of great importance nowadays. Within this paper, we propose a system modeling based approach for the integration and test of such systems. A V-model is introduced, targeting process oriented needs for safety and indicates where modeling languages in favor can be applied best. To establish a link between safety goals and the structure of simulation models, the initial model is enriched with necessary information and transformed to a language suitable for advanced simulation tasks. SystemC has the capabilities to support this approach for hardware and software even-handedly. The integration of SystemC into a co-simulation environment also enables the usage of external simulation models within the proposed architecture. The proposed system modeling based approach enables safety verification and validation at an early stage of development.

REFERENCES

1. International Organization for Standardization, “ISO 26262: Road Vehicles - Functional Safety - Part 1-10.” 2011.

2. Raslan W. and Sameh A., “System-level modeling and design using SysML and SystemC,” *Integrated Circuits, 2007. ISIC'07.*, pp. 504-507, 2007.

3. Prevostini M. and Zamsa E., “Sysml Profile for SOC Design and SystemC Transformation,” *ALaRI, Faculty of Informatics*, University of Lugano, 2007.

4. David P., Idasiak V., and Kratz F., “Improving reliability studies with SysML,” *Annual Reliability and Maintainability Symposium*, pp. 527-532, Jan. 2009.

5. Thramboulidis K. and Scholz S., “Integrating the 3+ 1 SysML view model with safety engineering,” *Emerging Technologies and …*, pp. 13-16, 2010.

6. Schneider, R., Brandstaetter, W., Born, M., Kath, O. et al., “Safety Element out of Context - A Practical Approach,” SAE Technical Paper 2012-01-0033, 2012, doi: 10.4271/2012-01-0033.

7. “OMG Systems Modeling Language (OMG SysML).” Object Management Group, 2012.

8. “IEEE Standard 1666: SystemC Language Reference Manual.” IEEE Computer Society for Accellera Systems Initiative, formerly known as Open SystemC Initiative, 2011.

9. Barnasconi M., “SystemC AMS Extensions : Solving the Need for Speed,” 2010.

10. Ip C. N. and Swan S., “A Tutorial Introduction on the New SystemC Verification Standard,” *Design, Automation and Test in Europe*, 2003.

11. “ICOS Independent Co-Simulation - User Manual Version 2.” The Virtual Vehicle Research and Test Center, Graz, Austria, 2011.

12. Hofmann P., *Hybridfahrzeuge : Ein alternatives Antriebskonzept für die Zukunft*. Springer-Verlag, 2011.

CONTACT INFORMATION

Martin Krammer
Virtual Vehicle Research and Test Center
Area E - Electrics/Electronics & Software
Inffeldgasse 21a/1
8010 Graz, Austria
Martin.Krammer@v2c2.at
http://www.v2c2.at

Helmut Martin
Virtual Vehicle Research and Test Center
Area E - Electrics/Electronics & Software
Inffeldgasse 21a/1
8010 Graz, Austria
Helmut. Martin@v2c2.at
http://www.v2c2.at

Dr. Michael Karner
Virtual Vehicle Research and Test Center
Area E - Electrics/Electronics & Software
Inffeldgasse 21a/1
8010 Graz, Austria
Michael.Karner@v2c2.at
http://www.v2c2.at

Dr. Daniel Watzenig
Virtual Vehicle Research and Test Center
Area E - Electrics/Electronics & Software
Inffeldgasse 21a/1
8010 Graz, Austria
Daniel.Watzenig@v2c2.at
http://www.v2c2.at

[6] http://www.avl.com

Dr. Anton Fuchs
Virtual Vehicle Research and Test Center
Area C - NVH and Friction
Inffeldgasse 21a/1
8010 Graz, Austria
Anton.Fuchs@v2c2.at
http://www.v2c2.at

ACKNOWLEDGMENTS

The research leading to these results has received funding from the ARTEMIS Joint Undertaking under grant agreement n° 295311, from the Austrian Federal Ministry for Transport, Innovation and Technology (BMVIT), the Austrian Federal Ministry of Economy, Family and Youth (BMWFJ) and the Austrian Research Promotion Agency (FFG).

DEFINITIONS/ABBREVIATIONS

ASIL - Automotive safety integrity level

ECU - Electronic control unit

E/E - Electrics/Electronics

HCU - Hybrid control unit

HARA - Hazard analysis and risk assessment

ICOS - Independent Co-Simulation

ISO - International organization for Standardization

OSCI - Open SystemC Initiative

SEooC - Safety Element out of Context

SysML - Systems modeling language

The Engineering Meetings Board has approved this paper for publication. It has successfully completed SAE's peer review process under the supervision of the session organizer. This process requires a minimum of three (3) reviews by industry experts.

ISSN 0148-7191

Positions and opinions advanced in this paper are those of the author(s) and not necessarily those of SAE. The author is solely responsible for the content of the paper.

SAE Customer Service:
Tel: 877-606-7323 (inside USA and Canada)
Tel: 724-776-4970 (outside USA)
Fax: 724-776-0790
Email: CustomerService@sae.org
SAE Web Address: http://www.sae.org
Printed in USA

SAE International®

Challenges for Reuse in a Safety-Critical Context: A State-of-Practice Study

2014-01-0218

Published 04/01/2014

Helmut Martin
Kompetenzzentrum Das Virtuelle Fahrzeug

Stephan Baumgart
Volvo Construction Equipment

Andrea Leitner and Daniel Watzenig
Kompetenzzentrum Das Virtuelle Fahrzeug

CITATION: Martin, H., Baumgart, S., Leitner, A., and Watzenig, D., "Challenges for Reuse in a Safety-Critical Context: A State-of-Practice Study," SAE Technical Paper 2014-01-0218, 2014, doi:10.4271/2014-01-0218.

Abstract

The need for cost efficient development and shorter time to market requires reuse of safety-critical embedded systems. One main challenge for reuse approaches in a safety-critical context is to provide evidence that assumptions of the safety artifacts for the reused component are still valid in the new system definition.

This paper summarizes the major findings from an explorative study conducted in order to identify the state of practice of reuse in the context of different functional safety standards. The explorative study consists of a set of questions, which have been discussed with interviewees from companies of various domains. The companies act in safety-critical domains with diverse product portfolios. We covered several points of view by interviewing persons with different background.

The results of the study reveal industrial challenges, which built the input for the derivation of possible future work based on the identified practical needs. Our main findings show the current predominance of ad-hoc reuse techniques and the need for more systematic approaches for reuse. We propose a systematic approach to cover the industrial challenges: establishing a safety culture in the company, an integrated system and safety development process, the introduction of model-based development for an improved support of reuse concepts, and metrics for impact analysis.

Introduction

Electrical and/or electronic (E/E) embedded system development is one of the main drivers for innovation, but still challenging in various domains. The need for cost efficient development and shorter time to market requires the application of reuse. Nevertheless, reuse has not only positive effects on the safety of a new system for example the reuse of a software component. The specific hazards of the new system were not considered when the reused software was designed and coded [1]. Safety is not a property of the software itself, but rather a combination of the software design and the environment where the software is used: So it is application-, environment-, and system-specific. Therefore, software which is safe in one system and environment may be unsafe in another.

Rigorous functional safety standards need to be applied in domains such as automotive or avionics, because malfunctions of E/E systems potentially cause hazardous events that could, in worst case, harm people. Functional safety aims to get a clear understanding of potential problems, their causes and effects, and provides possible fault solutions and mitigation measures to ensure a safe state in every possible hazardous situation.

For all functional safety standards it is vital that safety analysis results are updated whenever parts of the system or its operating conditions change. Repeating the entire safety assurance and certification would be valid, but also costly. In order to improve the development process it is therefore not only important to reuse development artifacts, but also to reuse their corresponding certification-related artifacts. Cross-domain reuse is another important application scenario. It means that a component certified for one standard shall be used in an industry domain applying another safety standard. Again, a complete recertification would be required despite the fact that most of the functional safety standards for E/E systems have a lot of commonalities.

The purpose of this explorative survey is to get insights into the state of practice for reuse and systematic reuse with product lines for software-intensive system in a safety-critical context. We want to understand which challenges arise in the context of functional safety and which best practices may already have been identified. Therefore, the following three research questions shall be answered by the help of structured interviews:

Research question 1: What are main challenges for the development of safety-critical products when (a) reusing artifacts and (b) using the software product line concept?

Research question 2: What are best practices for the development of safety-critical products when (a) reusing artifacts and (b) using the software product line concept?

Research question 3: How are product line concepts used in real industry cases?

This paper is structured as follows: Section 2 introduces some relevant standards and summarizes related work. Section 3 describes the structure, design, and rational of the questionnaire, how the interviews have been conducted, and how we have analyzed the results. Section 4 discusses the interview results. In Section 5, we derive the main challenges identified during the interviews and Section 6 summarizes the main findings from the interviews with respect to the previously defined research questions. The identified challenges are used as input for the definition of possible improvements. An outlook on this future work is given in Section 7. Section 8 finally concludes the paper.

Background and Related Work

The following paragraphs provide a short introduction to applicable specifications and functional safety standards from different domains.

The IEC/PAS 62814 [2] is an evolving public specification[1], which currently promotes reuse-driven software development. Two main aspects of component reuse are addressed:

(1) "Build-for-reuse" (planned production of reusable components) and (2) "Build-by-reuse" (planned production of systems using reusable components). This standard defines processes for both types of reuse and a combined development process.

The IEC 61508 "Functional safety of electrical / electronic / programmable electronic (E/E/PE) safety-related systems" [3] is a generic safety standard and is intended as a fundamental safety publication.

ISO 26262 "Road Vehicles - Functional Safety" [4] is a specific derivation of the IEC 61508 for the automotive domain. It has to be applied since November 2011 and covers the whole safety life cycle from development over production to service processes for E/E systems within road vehicles. The standard provides guidance to handle the increasing complexity of E/E systems and to reduce the risk of systematic development failures. Nevertheless, this standard comes with a big challenge for companies, because it sets requirements and prescribes a reference workflow, but does not explicitly explain how this can be implemented efficiently.

Concerning reuse it includes two means:

One of them is "proven in use argument"[2] which is an alternate means of compliance with ISO 26262 that may be used in the case of reuse of existing items or elements when sufficient field data is available.

The other one is "Safety Element out of Context (SEooC)"[3], a safety-related element which is not developed for a specific item (e.g. not developed in the context of a particular vehicle). A SEooC can be a system/subsystem, a software component or a hardware component. An example of SEooC for software is an AUTOSAR software component.

For the development of safety-critical software intensive systems within the avionics sector the DO-178B/C "Software Considerations in Airborne Systems and Equipment Certification" [5,6] defines guidelines for the specification, development, and verification of software components. Furthermore, it defines a software life cycle and its mapping to the corresponding system life cycle. The software inherits its integrity levels from the system integrity levels.

The ISO 15998 standard for the "Earth-moving machinery" [5] is defining safety requirements for heavy construction equipment machines. This functional safety standard is in turn pointing towards other standards like IEC 61508, ISO 13849 [8], and in the future to relevant parts of the ISO 26262 [4].

There has been an empirical study in a previous ARTEMIS JU project called CESAR[4] about how product line engineering (PLE) is used in industry [9]. In contrast to this survey, we will cover product lines and the more general reuse approaches, because the concept of PLE seems to be hardly known in companies developing safety-critical embedded systems. Due to the different scopes of the two surveys, we did not perform a detailed comparison of results.

The interview study has been conducted in the course of the ARTEMIS JU SafeCer[5] project. The project provides support for efficient reuse of safety certification arguments and prequalified components within and across industrial domains. One specific part of the project deals with platform

[1] Not a mandatory standard, but guidance which can be considered as a "pre" standard.

[2] See ISO 26262-8, Chapter 14 [4]

[3] See ISO 26262-10, Chapter 9 [4]

[4] www.cesarproject.eu

[5] www.safecer.eu

development and software product lines. This is why we are also interested in the application of this systematic reuse concept in practice.

It further aims to increase efficiency and reduce the time-to-market by the use of composable safety certification in several domains as automotive, construction equipment, avionics, rail, and health-care. Nevertheless, this paper will show the practical challenges of reuse in a safety-critical context to the research field.

Design of Interview Questionnaire

This section gives an overview on the design of the questionnaire structure, how we conducted the interviews and why we decided to do it this way. We also outline the experience and background of the interviewed persons and their respective company context. The mentioned research questions are of exploratory character and the interview design is based on Yin, et al. [10]. The questions of the study are available as part a public deliverable [11] at the SafeCer project website[6].

With this questionnaire we aim to gain knowledge about the state of practice for reuse in the context of functional safety in different domains. The questionnaire design is of an explorative character. We decided to use open questions which can be seen as a structured guidance through a conversation with the interviewee. We intentionally did not send out a survey to project partners with a set of predefined answers for each question. This strategy would have caused less effort, while giving us the possibility to get a much higher number of survey answers. Instead we decided to have an open discussion based questionnaire, which provides the possibility to get an immediate feedback from the interview partners, if their answers have been understood correctly.

The main advantage of discussions with interviewees is the possibility to get a better understanding of the specific processes, methods, and tools applied in the respective companies. Having a clearly defined questionnaire design helps to keep a common thread throughout the interview, while asking more detailed questions if necessary. Written answers, in contrast, are often short and hard to understand. One main prerequisite for a written survey would have been a clear definition of questions. Especially in a field with various terms describing basically the same concepts this is almost impossible to do.

The questionnaire consists of 50 questions, which are separated in 4 main parts. Its basic structure is illustrated in Figure 1. With Part 1 of the questionnaire we aim to get a characterization of the investigated companies and the respective interviewees. We want to understand their background, their daily work tasks, and their expertise within functional safety and reuse. This is required to interpret the answers given by the interviewees. Getting to know company-specific details helps us to understand why certain methods are applicable and when they are used. Additionally, we want to identify the domain and product portfolio of the respective company. This also includes questions about the development strategy, e.g. percentage of code developed in-house. The reason for this is that the development process in a company developing most of the code in-house might differ significantly from a company where lots of code comes from suppliers and has to be integrated. It is further important to get information about company-specific terminology, e.g. how they define the term product variant, if such variants exist in the company, and how they are handled. The same applies for product generations. These aspects are understood quite differently in companies and the impact on managing reuse and functional safety can differ accordingly. This first part of the questionnaire helped a lot in getting a clear understanding of the view and scope of the remaining questions.

In Part 2, we asked the interviewees for their personal definition for terms like functional safety, product lines/product families, and platforms. For us, this harmonization of terminology and understanding seemed to be very important. During the preparation of the questions we saw that the definition of the terms platform and product line can be different even in one company when asking people from different departments with different background and expertise. Especially the term platform is used differently in several contexts. So, for the successful and meaningful conduction of the remaining interview it is a main prerequisite to get a mutual understanding concerning the company specific terminology to perform a common evaluation of the results.

Part 3 and 4 of the questionnaire are alternative. Depending on the discussions during the interview one of these two question sets has been chosen. Part 3 covers reuse in general, whereas Part 4 includes some more detailed questions concerning the application of a software product line approach. We decided to have two alternatives in order to find out how companies deal with reuse in general or product line engineering in particular, if the interviewees are aware that their company uses a product line approach. The design process of the questionnaire has been iterative meaning that we tested each version of the questionnaire with various colleagues and tried to continuously improve the questions. Based on the results of the questionnaire we identify existing challenges and tried to derive possible improvements.

Interview Procedure

One big challenge in the interview process was to get access to appropriate interview partners in the various companies. First, it is not easily possible for an external person to know who the key persons are and second, it is not easy to get an appointment with those people. We got the first set of interview partners by asking our contact persons in the respective companies for recommendations. Then we introduced an additional question in the questionnaire, where we asked if the interviewee can recommend other interview partners which are able to give useful information.

[6] www.safecer.eu

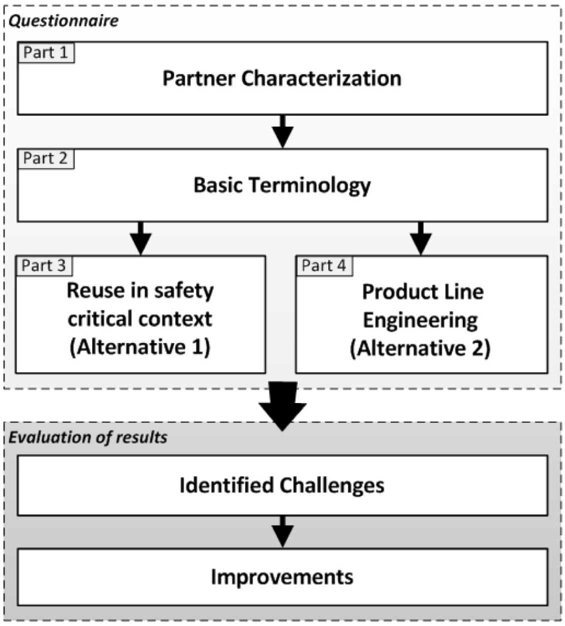


Figure 1. Overview of the structure of the interview questionnaire together with the evaluation of the results

With this strategy we could extend the set of interview partners. In order to be able to later analyze the results we wanted to record the interviews. This was not possible in all companies because of confidentiality issues. In cases where recording was not possible, two persons conducted the interview. One was carrying out the interview and the second one made notes. This was also very practicable, because it is not possible for one person to make notes while doing the interview. The actual interviews are very flexible. Some of the questions are already answered in the course of another question and can therefore be skipped. Sometimes it is better to change the order of questions, because of the current interview flow. During the meeting sessions we performed active listening to confirm what the interviewer has heard and moreover, to confirm the understanding of both parties. After the transcription the interview protocol containing all questions and answers was send out for a final review and that provides the possibility for additional remarks.

Analysis Procedure

We did not attempt to make a statistical evaluation of our results, because of the qualitative nature of the data. One main objective was to get some insights into current practices for reuse in a safety-critical context in order to identify challenges and open research questions. Therefore we mainly use the discussion to get an in-depth understanding.

Characterization of Interview Partners

The interviews have been conducted in 5 different companies which cover different domains and contexts. All of them are either developing safety-critical products or provide engineering services with scope on functional safety. Figure 2 shows the investigated companies in relation to their reuse strategy. Company 1 and 2 employ a product line concept for the development of safety-critical products. Company 3 is migrating from the development of products based on customer requirements towards a product line approach. Company 4 is providing engineering services specific for each customer, which makes systematic or planned reuse of artifacts more challenging. Company 5 is also moving towards a product line approach, since they are starting to combine common modules in a base system.

Generally we aimed to interview people with the scope on safety, software design and system design within each company. In the following we are giving a short overview on each studied case.

Case 1: Heavy Machinery Development

One set of interview partners is employed at a company that is developing a huge range of different heavy construction machines. Several product lines are maintained and in most products a set of functions is considered as safety-critical. Typical relevant functional safety standards are ISO 15998 [7], ISO 13849 [8], and partly IEC 61508 [3]. Several components are developed in-house, but a lot of parts are ordered and developed by suppliers. In this company, we interviewed a range of experts from different projects and areas. Accordingly we are able to identify how different product lines are developed and which challenges arise from different perspectives. The interviewees have been chosen according to their role in the company and dependent on their work tasks.

Case 2: Truck Development

As a second case study, experts and managers from a truck developing company have been interviewed. In the truck domain compliance to a functional safety standard is not yet required by legislation. Nonetheless, functional safety activities are common but are until today managed as quality assurance tasks. In order to capture the way of working and how product lines look like several responsible persons from different departments have been interviewed.

Case 3: Subsystem Supplier

For case 3, we interviewed a safety expert from a company that is supplying safety-critical subsystems to construction equipment and railway domain customers. The range of products reaches from non-safety-critical products to safety-critical products. Usually, the customer is defining the safety requirements and the company is tracing those requirements through development in order to show compliance when delivering the product to the customer.

Typical functional safety standards used in this case are EN-50128 [12], EN-50129 [13] and IEC 61508 [3]. Reusability and product lines play a more and more important role and a transition towards product line development is ongoing. The interviewee has been responsible for safety in several projects as a safety responsible.

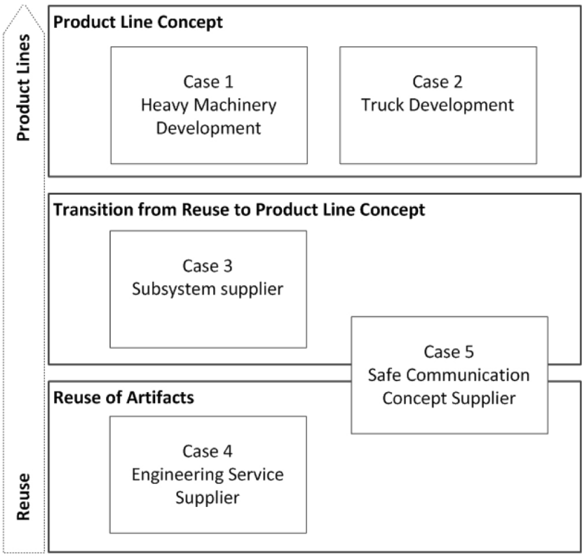


Figure 2. Characterization of interview partners according to their reuse strategy.

Case 4: Engineering Service Supplier

One of the interviewed companies is an automotive supplier which mainly provides different engineering services - in particular functional safety services. This company has to comply with different automotive standards for the development and in particular with functional safety according ISO 26262 [4]. They do different kinds of engineering services such as powertrain engineering, simulations, and software development for engines and transmissions. Reusability in this company is limited because of the customer-driven projects. We had interviewees from various departments and with different roles, e.g. project management, process responsible, safety engineer, and development engineers. All of them were very experienced in their specific position and could give as detailed insights in their everyday work.

Case 5: Communication Products Developer

The last company is a leading supplier of technology and software products in the field of time-triggered architectures and communication systems. Their products enable developers of aerospace, automotive, and off-highway industry to deliver more reliable embedded systems quickly and efficiently. Their products are successful in the aerospace industry and the company has gained wide experience in certification according RTCA DO-178B and DO-254 [14]. High ASIL levels (ISO 26262) are supported in projects with leading automotive industry partners. Furthermore the industry safety standard IEC 61508 is important for applications in that field, because it influences the development of E/E/PE safety-related systems and products across all sectors. The interviewees are at the one side responsible for the management activities in the company and in the coordination of different research and industry projects for aerospace and industry applications. In their department they have experiences with different safety standards (DO-178B/C, ISO 26262, IEC 61508) and the required safety process activities across these standards.

Questionnaire Results

This section summarizes the main results of the questionnaire and therefore uses the questionnaire structure as described in Section 3.

Terminology

First of all we are interested to know how the terms product line and platform are defined in an industrial context. This understanding is important for an investigation of reuse and product lines. Without a common wording it would be Impossible to interpret answers in a meaningful way. It further helps to identify if a company employs the product line concept without realizing it. This section describes our findings.

Product Line

The core idea of the product line approach is to build multiple products from a single infrastructure in a way that is aligned to stated business goals. An often used definition from Northrop and Clements [15] describes a software product line as “a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way.” In distinction to other reuse approaches, software assets themselves contain explicit variability. The main concept is the differentiation of domain and application engineering. In domain engineering, reusable assets are developed together with a description of the supported variability. Concrete products can be derived in application engineering using these reusable assets in a predefined way. The most important issue is the description of variability. This is usually done with variation points, which provide several possible variants, which can be chosen for a concrete product. At the moment a specific variant is selected, the variation point is said to be bound [16].

One major finding from our survey reveals that definitions do not only differ between companies, but also between departments within one company. The definition also depends on the role of the company, either as an OEM (Original Equipment Manufacturer) or supplier. OEMs typically define a product line as a group of products with some similarities that can be configured based on the requirements of a customer. Internal sub-products such as a common E/E architecture, engine steering or transmission steering are often defined as products of an internal product line. A product line is typically defined as a set of products that share a specific product characteristic, like type of machine or type of truck. Which products are seen to be sharing a specific characteristic is usually dependent on the company's product philosophy. Suppliers seem to face more challenges when employing a product line concept, since their products are often more dependent on specific customer requirements. In general, we can state that answers are very much dependent on the interviewees' tasks and working area in a company. As mentioned before, domain and application engineering are key concepts of software product lines in literature [18]. We saw that practitioners, at least in companies developing safety-critical products, are often not familiar with these terms.

Therefore it is hard to make a connection between the academic and the industrial world. Here, we also saw the advantage of conducting the survey as interviews. Otherwise interview partners simply would not have given answers to relevant questions, because they are not aware of the fact that they indeed use these concepts.

Platform

Basically, software product lines enable efficient mass-customization by the use of platforms. We use the following definition for platforms in the context of software product lines from Pohl et al [18]: "A platform is a base of technologies on which other technologies or processes are built". There are many other definitions for platforms available in literature. Most interviewees refer to the term software platform, which in their understanding consists of commonly used software modules. These software modules would not need to be redeveloped for each product and their reuse will reduce the development, verification, and safety assessment effort. Some interviewees stated that it can be difficult to reuse software components because of changing electronic hardware configurations or the need to migrate to another tool chain if the software platform should be applied in a new context. Therefore they use a much broader understanding of the term platform, which also includes common architectural guidelines, recommended tools, technical concepts, suggested hardware, software modules, and so on. With this enhanced view on platforms the potential for reuse and effort reduction is stated to be much higher. The verification effort for the common architecture, the tool chain, technical concepts and hardware is being reduced significantly. Again we found that terminology can be very different within one company. A platform can be seen on the one hand for example as the AUTOSAR [17] part of the steering system on an ECU. On the other hand a platform was described as a superset of all possible vehicle functions that can be build into a machine for a customer.

Generally we see that there is often no commonly defined, used, and harmonized terminology in companies and that the terminology used in practice is not compliant with the terminology used in literature. This makes it (1) hard to establish a company-wide culture regarding functional safety as well as reuse and (2) it hinders knowledge transfer from research towards industry and on the other side the identification of industry needs from an academic point of view. For both aspects, functional safety as well as reuse in general and product line engineering, it is important to have a clearly defined common understanding.

Reuse in a Safety-Critical Context

This section discusses reuse in a safety-critical context from two important points of view: the first part discusses reuse of software and other development artifacts, the second part focuses on reuse of safety artifacts.

Reuse of Software and Development Artifacts

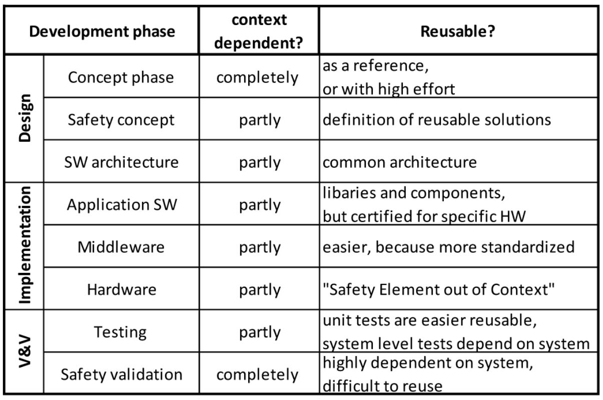
This part of the questionnaire asks for the degree of reuse, the type of reused artifacts, if and how much artifacts must be modified in order to be reused in a different context, and if there are dedicated methods for reuse. First, we observed that the need for reuse is highly dependent on the business strategy of the company. For one of the interviewed companies (Case 4) it is not important to force reuse. Their business strategy relies on the development of customer-specific solutions. They regard the know-how of their developers as their most valuable asset. But even for them it seems to be helpful to maintain a common software architecture for all products, which is then detailed with diverse concrete implementations. In this context it has also been reported that an advanced application life cycle management (ALM) tool can be an enabler for reuse. These tools support version control and branching and are often tightly integrated with the respective development environment. This enables the parallel development of different component variants and versions. Nevertheless, at least in the investigated company there is no systematic variability management support. The degree of reusability is also highly dependent on the specific domain. Some domains, as for example the aerospace domain (Case 5), have much more stable requirements as other domains. Sometimes it is even possible to reuse 100%. Other domains have very diverse products and therefore, it is hardly possible to reuse anything.

Reuse of Functional Safety Artifacts

The questionnaire contains a set of questions asking for reuse of functional safety artifacts (e.g. hazard analysis and risk assessment or functional/technical safety concepts), tool support, responsibilities, and argumentation in the context of reuse. Especially from a safety point of view results from previous projects are hard to reuse, because they are closely linked with the concrete project or the safety item, respectively. Just taking the example of a vehicle: Selecting a different type of transmission changes the whole safety concept starting from hazards and their assessment. This means that even for slightly different requirements, all safety artifacts have to be revised. Typical reuse techniques such as simple copy and paste are therefore hardly applicable for safety artifacts. Nevertheless, in some cases reuse is desirable at least for some parts. In a cross-domain setting, reuse of safety artifacts needs to match the new scope and the safety requirements in the different domains (Case 5). It is possible that a different kind of documentation is required for a different domain. E.g. a safety manual which documents the arguments for functional safety of delivered component is usually provided by the supplier in automotive domain. This kind of document is not known in the avionics domain.

General Discussion

Figure 3 summarizes our main findings about the relation between the different development phases and the potential for reuse. The picture is restricted to parts of the development process where we got answers from the interviews. It mixes up system/software and safety development and we do not claim this to be a complete development process. One column in the represented table states about the context-dependence of artifacts in this phase. We distinguish between completely and partly dependent. As we have described before, safety analysis is tightly connected to the specific system. This means that small changes in the system description require a complete rework of the safety analysis. If this is the case, we talk about completely context dependent artifacts. Otherwise, if it is somehow possible to separate the artifact from the current system context, we speak of partly context dependent artifacts. In the last column we indicate the reuse potential of artifacts in the respective development phase.



Development phase		context dependent?	Reusable?
Design	Concept phase	completely	as a reference, or with high effort
	Safety concept	partly	definition of reusable solutions
	SW architecture	partly	common architecture
Implementation	Application SW	partly	libaries and components, but certified for specific HW
	Middleware	partly	easier, because more standardized
	Hardware	partly	"Safety Element out of Context"
V&V	Testing	partly	unit tests are easier reusable, system level tests depend on system
	Safety validation	completely	highly dependent on system, difficult to reuse

Figure 3. Reuse potential in different phases of development

We do not claim this list to be complete because it summarizes the findings from our survey. In general, reuse seems to be most practicable in phases without complete context dependency. This is obvious because the artifacts are not tightly connected to a specific system description. Therefore, in the concept phase, previous project results are mainly used as a reference in a safety context. Hazard analysis and risk assessment is very subjective and usually requires some experience in order to identify the main hazards and to assess them appropriately. Previous hazard analysis and risk assessments can be used to cross-check the completeness if all hazards have been identified and can be used as a reference based on previous ratings. This is the main source of reuse of safety artifacts in the interviewed companies. The safety concept seems to be very context dependent as well, but it has been stated that it can be designed in a way that solutions are reusable in later projects as well. The structure of the software is also one reusable artifact as described above. Companies tend to employ a common software structure for all products if possible. The products then differ in the concrete implementation. Software seems to be easily reusable, because there are many concepts propagating software reuse, e.g. object-oriented development, component-based development, model-based development. In fact, in a safety context it is not that simple. Software can only be certified in combination with a concrete hardware platform. So it is not easily possible to reuse a software component on a different platform. Software as well as hardware can be developed as a Safety Element out of Context[7]. This basically means that it makes assumptions on the context and ensures the safety for an assumed context. This concept is often applied in practice if the context is not known at the beginning of the product development, which is a very common scenario during the development of embedded systems. More common is the reuse of more standardized parts, which are less dependent on specific customer requirements (i.e. it is easier to reuse communication technology than system-level concepts). This usually requires a detailed review of previous projects in order to find out the possibilities of reuse and to preempt hidden dependencies.

Product Line Engineering

This section summarizes our findings from the product line cases.

Management of Product Lines

In the first part of the product line question set, we asked for the management and application of product lines in an industrial context. One main observation revealed that domain and application engineering processes as defined in literature (e.g. [18]) are not explicitly defined in practice. Instead, platform and product development use quite similar development processes. It has further been reported, that the commonality and variability is not analyzed at the beginning of the product line activities as proposed in literature. This usually leads to challenges later in the project because variability and common parts are built on expertise and experience. We asked the interviewees to describe the company's development strategy in the scope of product lines. In one case, they developed the whole functionality of a high-end product and then used it as a base to derive other products. In this context it was reported, that the missing distinction between platform and product development processes is leading to problems as soon as the platform project ends, but products still use the platform. Additionally, development strategies change over time. In one case, the development strategy is evolving from product development based on customer requirements to the development of a customizable product base. In other cases attempts to add more functionality to the platform failed at the end. Possible reasons for this failure are a lack of understanding in the organization, unclear development strategies and management commitment. A product life-cycle lasting many years for products derived from the product line is challenging. This means that the product line needs to support both, future products but also products already existing on the market. It has been a general observation that maintenance of a product line is a big challenge.

[7] See detailed definition in ISO 26262-Part 10 [4]

Modeling of Product Lines

Model-based development is becoming more and more important for embedded system development. We have been interested in how models are used in real-case product lines and in the context of functional safety. Some steps towards UML modeling have been made in one case, but in general we could not find evidence for the use of modeling techniques. The design of a product line is usually based on experience and sketched in tools like MS PowerPoint. Architecture design languages like EAST-ADL or AADL are also not used in the investigated cases. Nonetheless, practitioners see a more structured way of designing the product line to be useful.

Challenges for Reuse in a Safety-Critical Context

The discussions with the interviewees revealed several challenges companies facing with reuse in a safety-critical context. Together with the interviewees we summarize the most important challenges and provide a common point of view:

Challenge 1: Context Dependency

As we have already discussed before, safety assessment is highly dependent on the concrete system context, which hampers the reuse of safety artifacts. One has to take care that assumptions on the system are still valid in the new context. Small changes in the overall system concept have a huge influence on the safety assessment of a system. Entering a new market or using a manual instead of an automatic transmission requires a completely new safety analysis and safety concept. Considering a new target market, the probability of a certain hazard in this environment (e.g. icy road in Africa) could be completely different.

Challenge 2: Uncertainty Due to Subjectivity - Experience Required

Risk assessment is highly dependent on the personal valuation of the given situation. The person doing this assessment has to find a trade-off between ensuring a safe system (which is his responsibility) and the costs of system development. The higher the assigned ASIL, the higher the costs. There are no given values how to rate the risk of a system in an objective way. This means that this task is highly dependent on the judgment and experience of the safety engineers. Furthermore, hazard analysis is a mainly manual task. It can be supported by tools and partly automated, but the main work has to be done by safety engineers. Therefore there is always some risk that someone oversees a potential hazard. Reference projects are a useful source for comparison to ensure that all possible hazards have been covered.

Challenge 3: High Complexity - Safety Artifact Dependency

The challenge of complexity is twofold: On the one hand, E/E systems are getting more and more complex because of the increasing number of functionality, especially cross-cutting functionality. This makes the analysis of possible faults very extensive and complex. On the other hand, safety analysis and modeling spans over the whole development process and results in a huge variety of documents. In order to argue the safety of a system, it is important to ensure traceability between these documents and the actual system implementation. This again introduces a lot of complexity.

Challenge 4: Integrated Development Process

In order to improve the development of safety-critical systems it would be valuable to have one common development process which is supported by an integrated tool chain. Often, safety is introduced at a very late point in the project. At this time, most design decisions have already been made and safety measures are costly to implement. Early consideration of safety in the development could safe efforts later on. A common development environment and process could support the joint, parallel development of technical and safety solutions as one product instead of trying to include safety into an almost complete product. The challenge of tool interoperability and integrated development is an ongoing research in different projects (e.g. ARTEMIS projects CRYSTAL or MBAT).

Challenge 5: Scope of Product Line Development

If a company wants to employ a product line, it is important to have a clearly defined scope from the beginning of the development process. In case of safety-critical product lines this does not only include the requirements for products and their variability, but also safety requirements and their possible variability. This means that reuse has to be defined at an very early stage of development. This can be challenging because safety introduces a lot more possible variability.

Challenge 6: Traceability and Documentation

As already has been mentioned above, safety is often seen as an “add-on” in late phases of development. This does not only make the solution expensive, but does also inhibit traceability. Traceability is required for safety argumentation and reuse. It is hardly possible to efficiently reuse parts of prior projects without making a detailed impact analysis. One important aspect is the documentation of design decisions from a very early stage of development. They are an essential part of the safety argumentation. If design decisions are linked to the respective solutions, the reusability of these solutions is also improved.

Challenge 7: Common Understanding

The development of a common understanding of the system and the applicable safety standard is an important prerequisite of the success of a project, especially for distributed projects. Safety requirements could be given by a customer or in other cases, the entire safety assessment is out-sourced to an external supplier. Anyway, distributed development of a system requires a clearly defined wording. Additional challenges are cultural differences. Due to subjective nature of safety analysis, the cultural aspect can lead to overestimation or under-estimation of a criticality of a function. This is challenging for global companies as well as in global organizations where development is distributed over different sites, the different views on safety criticality become a challenge.

Questionnaire Interpretation

In this section we summarize the main findings regarding the research question defined in the introduction.

Research question 1: *What are main challenges for the development of safety-critical products when (a) reusing artifacts and (b) using the software product line concept?*

In research question 1 we are aiming to collect such challenges. The identified challenges are not just of technical character, but are also organizational and related to the used development process. In fact the used product line concept was often described to be missing a clear process, good documentation, clear variant management and good change management. Traceability through all the safety-related artifacts is a vital issue to manage all changes and analyze the impact of these changes. Small changes in the system specification require a complete and expensive re-work of the safety artifacts. Organizational challenges like management support for product lines and functional safety or establishing a common understanding of used terminology require long term strategies. The company specific interpretation of functional safety standards should be done by experienced engineers that provide clear and unambiguous guidelines and workflows for the development team of safety-critical products. So it should be clear what and why they have to do it and how it can be done in an efficient way.

Research question 2: *What are best practices for the development of safety-critical products when (a) reusing artifacts and (b) using the software product line concept?*

Typically some interviewees started looking at model-based development to support their development work on the one hand and on the other hand to enable a safety analysis and safety assessment. In many cases we analyzed, the experts are working on aligning the development process with the relevant functional safety standard. It has been shown, that persons, which are involved in different projects, have different interpretations of the functional safety standards.

A common interpretation is therefore necessary. The establishment of a safety culture in a company is taking longer time and is affecting not only E/E development but also management, verification, production and even maintenance.

Research question 3: *How are product line concepts used in real industry cases?*

To our experience the product line concept described in literature [15] is not applied as is within the industrial context for developing safety-critical products. Instead product lines have developed over time within companies and different solutions have emerged. It is therefore necessary to distinguish between different product line solutions. In his paper [24] Jan Bosch describes different levels of maturity of software product lines. The author is providing categories of software product lines from very straight forward solutions where nothing is reused to very advanced solutions where evolutions of product lines are managed. This concept was our starting point for our study. From functional safety perspective each of the provided categories will result in different development solutions and therefore different ways to manage functional safety.

Accordingly we are interested first how product line solutions look like in industry and how the used concept evolved. Conducting and analyzing the interviews was challenging, if specific terms like platform and product line are used differently within a case and between different cases. The existing product lines have often developed over time and not through planning. For example in case 1 the separation of different product lines has its source in the acquisition of competitors with their product range. The platform used for most product lines is not derived from identifying commonalities and variabilities, but instead is a company decision to follow the AUTOSAR standard [17]. Some known commonalities have been added to this concept. This example shows how complex real-life cases are and how hard it is to gather right information that is possible to analyze and compare with others.

A set of characteristics needs to be set up to be able to understand the way of using the product line concept and to be able to collect data that is on the one hand comparable and on the other hand can be matched to concepts given in literature to evaluate their correctness. Asking for concepts described in literature like for example “domain engineering” and “application engineering” is only confusing and is not leading to answers. Instead characteristics that help to deduce if a specific process like domain engineering is applied, need to be used in the interviews. One important experience for us was the interesting insight in different aspects in various companies and departments.

Improvements and Future Work

Based on the answers from this questionnaire, we derived four possible fields of improvement regarding reuse and functional safety. These suggested improvements will be the base for the definition of concrete tasks for the second part of the SafeCer project. Figure 4 maps these improvements to the challenges identified in the previous section. It shows that we can cover quite a lot of the identified challenges in this way. In the remainder of this section we describe the suggested improvements in more detail.

Improvement 1: Establishment of a Model-Based Development Environment

Development engineers set a lot of hope in model-based development and the integration of safety in a model-based development environment. The system modeling approach presented by Habli et al [19] examine how model-driven development and assessment can provide a basis for the systematic generation of functional safety requirements. They demonstrate how an automotive safety case can be structurally and traceable developed, justifying why and how the defined functional safety requirements can adequately mitigate the risk of the identified hazards to an acceptable level. Kelly [20] proposes a modular, compositional approach for the construction of safety cases. Contracts are used to describe the agreed relationship.

An integrated system model supports different views on the specification of the safety-related system and improves the cooperation of involved disciplines in a company and the communication to their suppliers. Furthermore the support of traceability over the different safety artefacts is addressed by MBD in an efficient way. The component-based development and modularization is one vital contribution of the SafeCer project to extend this approach for safety-critical applications. It seems to be promising and a key to success.

Improvement 2: Integration of Safety in the Development Process

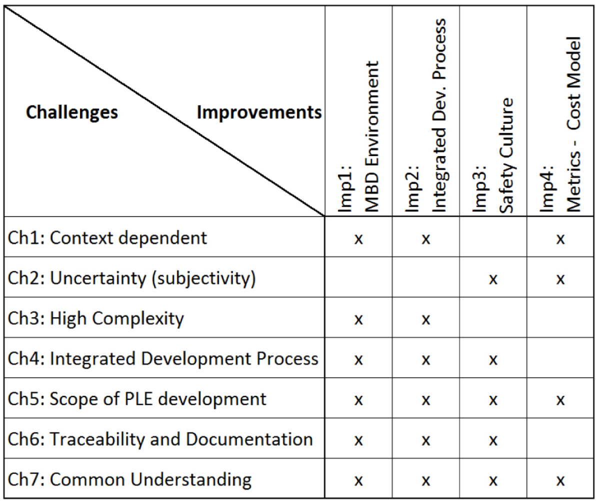
Well-established development processes are a valuable basis for the integration of the safety activities required by the different functional safety standards for the different domains. The investigation of an applicable integrated process model for domain-specific and cross domain deployment will also be one of the research topics in SafeCer. One further extension of that integrated process model shall be a work-flow, which supports the possibilities for introducing reuse of the elaborated safety artifacts.

Research in the field of safety-critical product lines has focused on adapting traditional safety analysis techniques, such as Fault Tree Analysis (FTA) and Failure Mode and Effects Analysis (FMEA), to suit product line processes. Much of the work has been conducted at the Laboratory for Software Safety at Iowa State University. Most noticeable is the extension of Software Fault Tree Analysis (SFTA) to address the impact of product line variation on safety analysis [21,22,23]. Furthermore the integration of the significantly involved development tools to an integrated and process oriented tool chain will be investigated in SafeCer projects.

Improvement 3: Establishment of Safety Culture

In most of the investigated companies, safety is regarded as a required task which can be handled by an independent safety team. Often engineers ask for support from management which should force a basic safety culture in their company. This means that the required safety activities should be regarded as an integral part of the development process and should not be in a parallel independent side rail. All persons, which are involved in any safety activity, have to be aware about their role in the organization and the project, their responsibilities and their expected contributions.

This further means that tool chains should be adapted in a way that they force engineers to perform certain tasks in close cooperation with the safety engineers. E.g. it seems to be desirable for software developers to get more training to improve the understanding and awareness for the topic of functional safety and the possibilities of application of the approaches for reuse. The SafeCer project is offering trainings and workshops at different industrial and scientific conferences and elaborates guidelines to support the different safety activities which are required for the certification of safety-related systems.



Challenges / Improvements	Imp1: MBD Environment	Imp2: Integrated Dev. Process	Imp3: Safety Culture	Imp4: Metrics - Cost Model
Ch1: Context dependent	x	x		x
Ch2: Uncertainty (subjectivity)			x	x
Ch3: High Complexity	x	x		
Ch4: Integrated Development Process	x	x	x	
Ch5: Scope of PLE development	x	x	x	x
Ch6: Traceability and Documentation	x	x	x	
Ch7: Common Understanding	x	x	x	x

Figure 4. Identified challenges and possible improvements

Improvement 4: Metrics for Impact Analysis

The introduction of reuse metrics will provide a quantitative indicator, which helps to evaluate the applicability, meaningfulness, and practicability of component reuse. The reuse metrics can be used to decide whether or not a component can be reused without modifications or if modifications are meaningful from an economical point of view.

SafeCer will investigate on a cost model for reuse that provides such kind of metrics, which should provide the basis for an objective decision. The cost model would first assist to identify and compare the best potential reuse candidate and second support the change impact, if requirements of the system changes to assess their effect on the existing reusable safety architecture by performing a quantitative impact analysis.

Future and on-going work of the SafeCer project will extend the methodology for different reuse approaches for the safety-critical context and provide guidance, how the potentials for reuse can be developed in an efficient way.

Conclusion

High development complexity, the need for shorter time to market, and cost efficient development of safety-critical embedded systems in domains like automotive, avionics and heavy machinery requires more sophisticated reuse strategies. These strategies are often not employed in practice. With this explorative study we provided an overview of current state of practice concerning reuse in the context of functional safety. From the results of the explorative survey we derived the main challenges and we suggest the following possible improvements as future work to overcome these challenges: Establishment of safety culture in the company, introduction of an integrated system and safety development process, and applying model-based development, which supports a systematic basis of reuse concepts, and metrics for impact analysis.

As a possible extension of the study it would be good to investigate other domains, like aerospace or healthcare, as well to gain more insights of industrial reality of reuse. A next step could be the elaboration of additional online questionnaire to include a more significant number of companies of each domain. This enables a statistical evaluation of the presented results for state of practice concerning reuse.

References

1. Leveson N.. Safeware: system safety and computers. ACM, 1995.
2. IEC62814/Ed1- Dependability of Software Products Containing Reusable Components - Guidance for Functionality and Tests (DPAS) Voting terminated 2012-09-14). International Electrotechnical Commission, 1 edition, 2012.
3. IEC61508 - Functional safety of electrical/electronic/programmable electronic safety-related systems. International Electrotechnical Commission, 2 edition, 2010.
4. ISO 26262:2011 Road vehicles - Functional safety International Standard (Parts 1-10). International Organization for Standardization, 1 edition, 2011.
5. RTCA DO-178B Software Considerations in Airborne Systems and Equipment Certification. RTCA, Inc." 1992.
6. RTCA DO-178C Software Considerations in Airborne Systems and Equipment Certification. RTCA, Inc." 2011.
7. ISO 15998:2008 Earth-moving machinery - Machine-control systems (MCS) using electronic components - Performance criteria and tests for functional safety. International Organization for Standardization, 2008.
8. ISO 13849 - Safety of machinery - Safety-related parts of control systems. International Organization for Standardization, 2006.
9. Schmid R.. State-of-the-art survey for product lines. Technical report, CESAR consortium, 2009.
10. Yin R. K.. Case Study Research - Design and Methods, volume 5 of Applied Social Research Methods Series. Sage Publications, Inc, 5 edition, 2008.
11. Baumgart Stephan, et al., Platform guidelines to support development projects in the context of product lines, Deliverable report pSC D2.1.2, Approaches of variant management in development of safety-relevant embedded systems implemented in nSafeCer meta-model, Deliverable report nSC D122.1 April, 2013.
12. EN-50128 Railway applications - Communication, signalling and processing systems - Software for railway control and protection systems. CENELEC - Comité Européen de Normalisation Électrotechnique, June 2011.
13. EN-50129 Railway applications - Communication, signalling and processing systems - Safety related electronic systems for signalling. CENELEC - Comité Européen de Normalisation Électrotechnique, February 2003.
14. RTCA DO-254 Design Assurance Guidance for Airborne Electronic Hardware. RTCA, Inc" 2000.
15. Northrop L. and Clements P.. Software Product Lines: Practices and Patterns. Addison Wesley, 2002.
16. van der Linden F. J., Schmid K., and Rommes E.. Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering. Springer, Berlin, 2007.
17. www.autosar.org Official website of the AUTOSAR partnership. Autosar.
18. Pohl K., Böckle G., and Van Der Linden F.. Software product line engineering: foundations, principles, and techniques. Springer-Verlag New York Inc, 2005.
19. Habli, I., Ibarra, I., Rivett, R., and Kelly, T., "Model-Based Assurance for Justifying Automotive Functional Safety," SAE Technical Paper 2010-01-0209, 2010, doi:10.4271/2010-01-0209.
20. Kelly T. P.. Concepts and principles of compositional safety case construction. Technical report, University of York, 2001.
21. Dehlinger J. and Lutz R. R.. Software Fault Tree Analysis for Product Lines. In 8th IEEE International Symposium on High-Assurance Systems Engineering (HASE 2004), pages 12-21, 2004.
22. Dehlinger J. and Lutz R. R.. PLFaultCAT: A Product-Line Software Fault Tree Analysis Tool. Autom. Softw. Eng., 13(1):169-193, 2006.
23. Feng Q. and Lutz R. R.. Bi-directional safety analysis of product lines. Journal of Systems and Software, 78(2): 111-127, 2005.
24. Bosch J.. Maturity and evolution in software product lines: Approaches, artefacts and organization. In Proceedings of the Second International Conference on Software Product Lines, SPLC 2, pages 257-271, London, UK, UK, 2002. Springer.

Acknowledgement

The research leading to these results has received funding from the ARTEMIS Joint Undertaking under grant agreements pSafeCer n° 269265 and nSafeCer n° 295373, Vinnova and the KKS-funded ITS-EASY Post Graduate School for Embedded Software and Systems. The authors from Austria would like to acknowledge the financial support from Austrian Research Promotion Agency (FFG) of the Federal Province of Austria for the project in which the above presented research results were achieved.

ISSN 0148-7191

http://papers.sae.org/2014-01-0218

Modeling a Safety- and Automotive-oriented Process Line to Enable Reuse and Flexible Process Derivation

Barbara Gallina and Shaghayegh Kashiyarandi
IDT, Mälardalen University,
Västerås, Sweden

Helmut Martin and Robert Bramberger
VIRTUAL VEHICLE Research Center,
Graz, Austria

***Abstract*—ISO 26262 is a recently introduced automotive functional safety standard. This standard imposes new requirements that must be fulfilled for conformance purposes. Thus, companies used to develop safety-related E/E systems in compliance with either only Automotive SPICE (ASPICE) or a combination of ASPICE and IEC 61508 have to quickly perform a gap analysis in order to introduce adequate changes in their way of working. Implementing such changes in a visionary way with expectations of a long-term payback is an urgent open issue. To contribute to addressing such issue, in this paper, we introduce a safety-oriented process line-based methodological framework to model commonalities and variabilities (changes) between the standards to enable reuse and flexible process derivation. To show the usefulness of our approach, we apply it to model a process-phase line for the development of safety-critical control units. Finally, we provide our lessons learned and concluding remarks.**

Keywords—Automotive SPICE; IEC 61508; ISO 26262; safety processes; safety-oriented process lines; process line modeling.

I. Introduction

In the context of safety-critical automotive systems engineering, various standards (e.g. ASPICE [1] and IEC 61508 [2]) play a crucial role in prescribing process reference models, which in some cases overlap and thus exhibit several similarities. More recently, ISO 26262 [3], which is a new standard for functional safety and which represents the automotive specialization of IEC 61508, has entered the scene with new requirements on the development process. As a consequence, since compliance with the process reference model may constitute a mandatory requirement for certification purposes, companies used to develop safety-related E/E systems in compliance with either only ASPICE or a combination of ASPICE and IEC 61508 have to quickly perform a gap analysis in order to introduce adequate changes in their processes. Via a gap analysis ad-hoc adjustments can be performed. Since however a gap analysis represents a timing window during which process engineers have to identify what changes and what remains the same, we propose to systematize this effort by implementing the safety-oriented process line approach, which was explored in [5] in the framework of the SafeCer [6] project. We thus propose to systematically model process elements as either commonalities or variabilities in order to enable reuse and flexible process derivation. The relevance of the adoption of a safety-oriented process line approach is motivated also by the fact that process reference models included in the standards allow flexible but thoroughly justified interpretations and customizations, which can be modelled as variabilities.

Thus, in this paper, to enable process engineers to implement changes with expectations of long-term payback, we introduce a methodological framework to model the commonalities and the variabilities that exist between automotive standards as well as the variabilities stemming from company-specific as well as project-specific interpretation and customization. Our methodological modelling framework uses the SPEM (Software Process Engineering Meta-model) 2.0 [8] process modeling language, which is implemented within the process modeling tool EPF-Composer [9]. To show the usefulness of our approach, we apply our framework to model a process-phase line for the development of safety-critical control units.

The rest of the paper is organized as follows. In Section II, we provide essential background information. In Section III we present our safety-oriented process line-based methodological modeling framework and its application. In Section IV, we draw some lessons learned. In Section V we discuss related work. Finally, in Section VI we present some concluding remarks and future work.

II. Background

In this section, we briefly present the background on which we base our work. In particular, in Section II.A we provide essential information concerning the system design phase of the automotive standards under examination. In Section II.B, we recall safety-oriented process line concepts and guidelines and, finally, in Section II.C we recall how processes and their variations can be modelled in SPEM2.0/EPF.

A. System design phase in the examined automotive standards

In this subsection, we focus on three standards, namely quality development standard ASPICE and two safety-related standards ISO 26262, and IEC 61508. The rationale behind the selection of these three standards is that they represent different but overlapping intra-domain (namely, automotive and industry) standards. For each standard, we provide a brief overview and then we focus on the system design phase. For this phase, we provide pointers to the normative parts, which are necessary to fully understand the application of our safety-oriented process line-based approach presented in Section III.

IEC 61508 deals with generic functional safety and is intended as a basis safety standard, applicable to different domains. ISO 26262 is one derivation of the IEC 61508. These two standards provide prescriptive development processes for achieving functional safety by reducing the risk of systematic failures. ISO 26262 and IEC 61508 include quality requirements that are in common with ASPICE, which focuses on process improvement. ASPICE provides a Process Reference Model (PRM). The PRM is composed of activities (in ASPICE named *Base Practices*), which are also covered by ISO 26262 and IEC 61508.

In ASPICE the majority of the activities concerning system design is part of the process *ENG.3 System architectural design*. Within this process the *Process Purpose*, the *Process Outcomes* and *Output Work Products* are defined. The structure of IEC 61508 differs significantly from ASPICE. IEC 61508 does not describe what should be done but sets out objectives and requirements for the activities. The system design phase is covered mainly in chapter 7.2 *E/E/PE system design requirements specification* and 7.4 *E/E/PE system design and development* of part two. Work products are not mentioned in the normative part. Finally, ISO 26262 part 4 – chapter 7 covers system design (entitled *System design*). This chapter includes objectives, input from other activities, requirements for the activities and work products.

B. Safety-oriented process line: concepts and guidelines

A *process line* [4] is a family of highly related processes that are built from a set of core process assets in a pre-established fashion. A *safety-oriented process line* is a process line that targets safety processes [5]. A (safety-oriented) process line approach is constituted of three phases: scoping (i.e. definition of the set of processes to be examined as a family), domain engineering (i.e. commonalities and variabilities identification and modeling), process engineering (single processes modelling via selection and composition of reusable commonalities and variabilities). Comparisons among safety processes characterize the main activity of the domain engineering phase. Through comparisons, it is possible to identify what can vary (variabilities) between safety processes and what, instead, remains unchanged (commonalities). At a first glance, processes defined in different standards seem to exhibit only variabilities. Terminological differences constitute a barrier to a straightforward identification of commonalities. Moreover, processes are constituted of phases, which in turns are constituted of a set of activities, which in turn are constituted of a set of tasks and which, finally, in turns are constituted of a set of steps. Thus, commonalities are unlikely at the root level of this nested structure. From an execution point of view, phases, activities, tasks, etc. may be performed in a different order. From a pure syntactical comparison, all these differences may be interpreted as variabilities. However, to be able to justify a process line approach the amount of commonalities must be greater than the amount of variabilities. To reduce the variabilities and increase the commonalities, two definitions are at disposal:

Partial commonality: whenever process elements of the same type (e.g. tasks) expose at least one common aspect (e.g. at least a step is equivalent).

In this paper, this definition is used in a flexible way. When comparing process elements of the same type, either the entire set of processes (process line) is considered or subsets of them. More specifically, the heterogeneous set of standards examined in this paper is divided into two subsets: one containing the non-safety-related standard (ASPICE) and the second containing the safety-related standards (ISO 26262 and IEC 61508). This flexible usage provides the potential to create a greater extent of reusable process elements.

Full commonality whenever process elements of the same type (e.g. tasks) expose only common aspects (e.g. all steps are equivalent).

For the sake of terminological completeness, we also clarify that a process variant is a representation of a particular instance of a variable process element of the real world or a variable property of such an element.

C. SPEM2.0 and Eclipse Process Framework

SPEM 2.0 [8] is the OMG (Object Management Group)'s standard for software process modelling. In the literature, several process modelling languages are available [10-12]. SPEM 2.0 is simply one of them but since it has appealing features in terms of standardization, reuse, tool-support, etc. (as surveyed in [12]) as well as in terms of active community working towards its enhancement [13], it answers our expectations. SPEM 2.0 offers static as well as dynamic modelling capabilities, the latter achieved by including links to other modelling languages (e.g. UML activity diagrams). SPEM 2.0 also offers modelling capabilities to address process variability. As explored in [5] these modelling capabilities are not fully adequate to model process lines. However, the alternative modelling proposal [14], called vSPEM, which is currently matter of investigation, is still too immature to be considered in the time-frame of our project.

In SPEM 2.0, a process element (e.g. a task) can be a variability element and to it the process engineer can associate separate objects representing the differences (e.g. additions) relative to the original (called base). The variability element has an attribute that characterizes its variability type. The variability type enumeration class defines the different types of variability [8]. In Table 1, we recall those variability types used in our approach presented in Section III.

TABLE I. VARIABILITY TYPES IN SPEM2.0

Variability type	Description
na	Not applicable
contributes	Provides a way to contribute to attribute values and association instances of the base, without altering it. The base is logically replaced with an augmented variant.
replaces	Defines a replacement of a base. The replacement consists of either a complete new variant or a change concerning fundamental relationships.

SPEM2.0 is defined as a Meta Object Facility (MOF)-based meta-model, which is composed of seven main packages, which are: 1) The *Core* package defines concepts allowing for the foundation of the other packages. 2) The *Method Content*

package defines concepts allowing for the specification of a knowledge base of reusable process elements (e.g. TaskDefinition and WorkProductDefinition). 3) The *Process Structure* package defines concepts allowing for the representation of process models composed of inter-related actual process elements (e.g. TaskUse). 4) The *Process with Method* package defines concepts such as Method Content Use elements (e.g. TaskUse) for the integration of processes defined by using the concepts available in Process Structure with the instances of concepts available in Method Content. 5) The *Method Plugin* package defines mechanisms allowing for the reuse and management of method content and processes. 6) The *Process Behaviour* package defines mechanisms and concepts allowing process elements to be linked to external models for behavioural specification. 7.) The *Managed Content* package introduces concepts for managing the textual content of natural language descriptions, which are necessary to increase models understand-ability.

For a subset of the concepts (that belong to the meta-model), graphical modelling elements (icons) are at disposal. Table II recall those icons that are related to what is presented in Section III.

TABLE II. ICONS DENOTING METHOD CONTENT (USE) ELEMENTS

Task Definition	TaskUse
Method Content	***Method Content Use***

Concerning tool-support for SPEM2.0, EPF Composer [9] is a tool that provides sufficient support with respect to our exploratory needs.

III. SAFETY- AND AUTOMOTIVE-ORIENTED PROCESS LINE

As discussed in the background section, to be able to reuse process elements, a process line approach is beneficial since commonalities and variabilities can be clearly systematized. As mentioned, currently, there is no satisfying modelling language and no tool supporting process lines. However, SPEM2.0 and EPF Composer are sufficient to implement our safety-oriented process line-based methodological framework and show its validity. Thus, in Section III.A, we illustrate how commonalities and variabilities can be modelled (domain engineering phase), by using SPEM2.0/EPF, and then, in Section III.B, we exploit those commonalities and variabilities to derive standard-specific single processes (process engineering phase). Fig.1 depicts the two phases and their main activities.

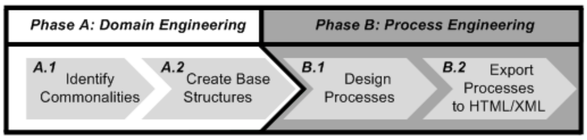

Phase A: Domain Engineering
Phase B: Process Engineering
A.1 Identify Commonalities
A.2 Create Base Structures
B.1 Design Processes
B.2 Export Processes to HTML/XML

Fig. 1. Process line phases

In the first phase we focus on the static structure of the process line. No modelling support is currently available to consider variabilities in the execution order. With respect to single processes, instead, we consider both static and dynamic structures.

A. Domain Engineering

The goal of this subsection is to explain how commonalities and variabilities are identified and modelled within SPEM2.0/EPF.

Before a process line model can be built, since all the three standards use slightly different terms to denote process elements (e.g. tasks, work products, etc.), a common terminology for all the three standards has to be defined in order to go beyond irrelevant terminological differences (Fig. 1.A.1). Development of semantic equivalence is a vital building step in process line creation. This step, which is extremely time-consuming, should be performed in close cooperation with experienced assessors to guarantee the achievement of a certifiable result. The challenge of this step is the identification and comparison of different terms within unstructured text fragments. For example, as summarized in Table III, the term *Base practice* in ASPICE is used as equivalent to the term *Activity*. Once the process elements (e.g. activities) got an identifier the mapping of different standards takes place whereby tasks are defined.

TABLE III. MAPPING OF SPECIFIC TERMS

Common identifier	ISO 26262	IEC 61508	ASPICE	EPF-C
Activity	Activity	Activity	Base practice	Task

If a process element (e.g. an activity) of one standard (e.g. *System design specification* in ISO 26262) is equivalent with a process element of a different standard (e.g. *E/E/PE system design and development* in IEC 61508) the elements are mapped to a common identifier. A unique ID is given to each matching comparison in order to provide traceability to the origins of all standards and the information is collected in a spreadsheet, which allows for tracking each single element and ensuring full coverage of the standards. Additional details can be found in [18]. After having identified the commonalities and variabilities that characterize the system design phase mandated by the set of standards considered in the background, SPEM/EPF can be exploited to create a knowledge base populated by those identified commonalities and the variabilities. As recalled in Section II.C, SPEM2.0 offers a package (*Method Plugin*) that supports reuse and management of method content. Within EPF this package is implemented and thus it is possible to create plug-ins containing reusable method content. As depicted in Fig. 2, we thus decide (as it was initially proposed in [17]) to define a series of plug-ins aimed at containing base elements. We organize these plug-ins by using two logical packages (*Base* and *Processes*), which in EPF Composer are aimed at grouping method plug-ins. We use Base (respectively Processes) for organizing plugins related to the Domain (Process) engineering phase. More specifically, we define one plug-in for each type of commonality (either full or partial) and variability (i.e. optional). We also define a plug-in for all the variants that are related to either partial commonalities or variabilities.

More precisely, as Fig. 2 shows, base elements include:

Full commonality: In the defined scope (see II.A) this type of element is very rare. Therefore partial commonality is used in a flexible way.

1) Partial commonality: Elements that to be reused need to be enriched in an additive way. These elements contain a common part. For example, a new task obtained by considering the subset of steps that is in common either in all three standards or in a subset of them.

As an example of elements of type "partial commonality", we can consider the task *Define system architectural design*. This task is present in all three standards. In ISO 26262 it is called "System design specification", in ASPICE it is called *Define system architectural design* and, finally, in IEC 61508 it is called *E/E/PE system design and development*.

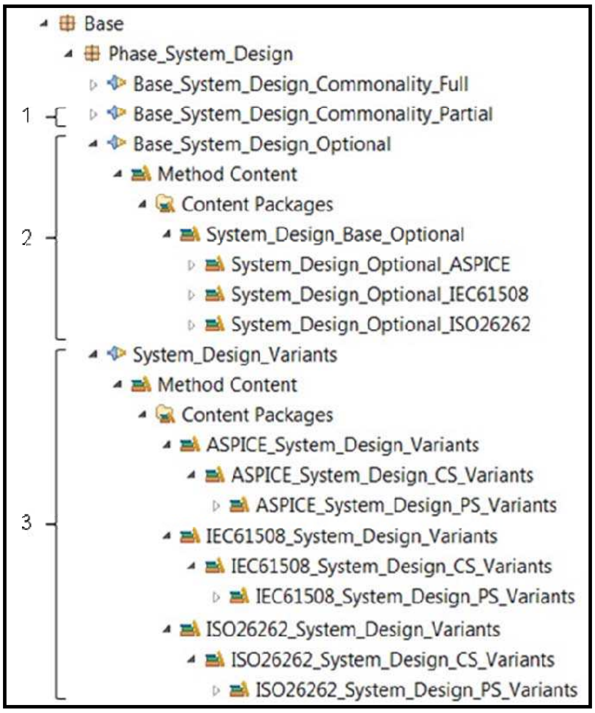

Base
Phase_System_Design
Base_System_Design_Commonality_Full
1
Base_System_Design_Commonality_Partial
Base_System_Design_Optional
Method Content
Content Packages
2
System_Design_Base_Optional
System_Design_Optional_ASPICE
System_Design_Optional_IEC61508
System_Design_Optional_ISO26262
System_Design_Variants
Method Content
Content Packages
ASPICE_System_Design_Variants
ASPICE_System_Design_CS_Variants
ASPICE_System_Design_PS_Variants
3
IEC61508_System_Design_Variants
IEC61508_System_Design_CS_Variants
IEC61508_System_Design_PS_Variants
ISO26262_System_Design_Variants
ISO26262_System_Design_CS_Variants
ISO26262_System_Design_PS_Variants

Fig. 2. Plug-ins for systematizing process line elements

The clauses below provide in italics the associated common steps, which justify the identification of a partial commonality:

ISO 26262: "The consistency of the preliminary architectural assumptions in ISO 26262-3:2011, 8.3.2 and the preliminary architectural assumptions in this sub-phase shall be ensured" [3]. Architectural assumptions are those related to the *representation of the structure of the item or functions or systems or elements that allows identification of building blocks, their boundaries and interfaces, and includes the allocation of functions to hardware and software elements*.

ASPICE: "Establish the system architectural design that *identifies the elements of the system* with respect to the functional and non-functional system requirements."[1]

IEC 61508: "The design shall be based on a *decomposition into subsystems with each subsystem having a specified design* and set of integration tests." [2]

2) Optional elements: Elements that might be standard-specific and that do not represent a mandatory element for each process of the process line. Optional elements can be replaced by an empty element if the single process to be derived from the process line does not include it.

The process element *Communicate system architectural design* is optional because it is only considered in ASPICE.

3) Variant elements: Base elements also include reusable standard-specific variants. These elements are named as variants in Fig. 2 and they are obtained by enriching the elements representing partial commonalities. For the complete creation of a process line for each process element (e.g. task, work product, guideline, etc.), several variants (e.g. standard-specific, company-specific, project-specific, etc.) should be provided (Fig. 1.A.2). Typical standard-specific variants are those that deal with different safety integrity levels (SIL or ASIL). Thus, the plug-in named with the suffix *Variants* also includes process elements that are not predetermined by a standard. For the creation of company-specific (CS in Fig. 2) as well as project-specific (PS in Fig. 2) process elements, standard-specific partial commonalities should be enriched or replaced (via the *contributes* or *replaces* relationship; see Table I).

As an example of elements *of* type "company-specific variant elements", we can consider the task named *Define system architectural design_V_SafeCer* that replaces the base element of type "*partial commonality" named Define system architectural design* with a SafeCer specific system architecture activity. The *description* field associated to each process element is then filled in with brief information as well as pointers to the spreadsheet containing the references to the sections of the single standards.

B. Process engineering

In the previous subsection we have shown how the domain engineering phase (phase A of the process line engineering) can be performed within SPEM2.0/EPF. To proceed with phase B, we create a new plug-in for each single process that belongs to our process line and can be obtained by selecting and composing process elements contained in the base (Fig.1 B.1). Fig. 3 shows the plug-ins that should be available at the end of the process line engineering development.

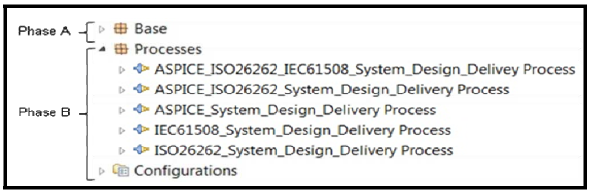

Phase A
Base
Processes
ASPICE_ISO26262_IEC61508_System_Design_Delivey Process
ASPICE_ISO26262_System_Design_Delivery Process
Phase B
ASPICE_System_Design_Delivery Process
IEC61508_System_Design_Delivery Process
ISO26262_System_Design_Delivery Process
Configurations

Fig. 3. The two-phase process line engineering approach

More specifically, the package "Processes" is expected to contain a plug-in for each single process of interest. Before a plug-in for a delivery process is created the area of interest has to be defined (e.g. ISO 26262). The specific process (e.g. *ISO26262_System_Design_Delivery_Process*) is built up with base tasks and the variants of the base tasks, which are provided in the *Base* package. For each delivery process a new plug-in is created where only the subsection *Delivery Process* is used. The creation of the real process occurs in the *Work Breakdown Structure* by using the tasks from the repository. To create an *Activity Diagram* related to the process the single tasks have to be properly ordered by setting their predecessors.

To create a delivery process concerning ASPICE and ISO 26262 (*ASPICE_ISO26262_System_Design_Delivery_ Process*), it is necessary to add all the needed process elements. This means that all elements in the method content *Full* and *Partial* must be added to the *Work Breakdown Structure*. Which elements from the method content *Optional* are added depends on the considered standards. If the focus is on the combination of ASPICE and ISO 26262 the elements concerning these two standards have to be added. At this point a delivery process is established which is compliant with two desired standards. Fig. 4 shows the corresponding activity diagram.

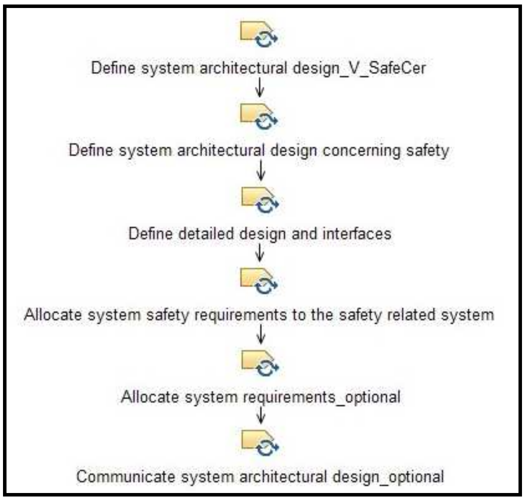
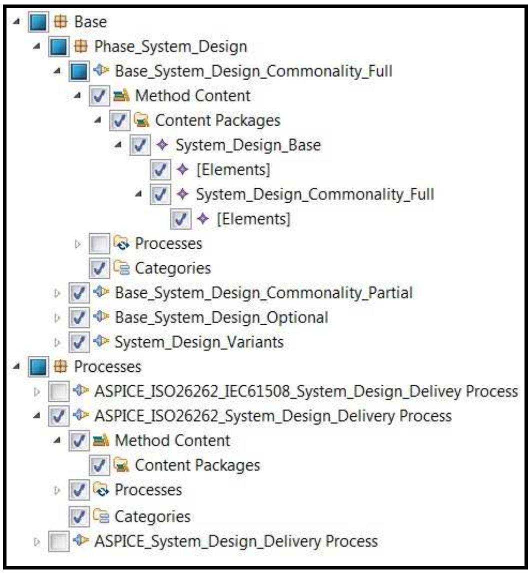

Define system architectural design_V_SafeCer
Define system architectural design concerning safety
Define detailed design and interfaces
Allocate system safety requirements to the safety related system
Allocate system requirements_optional
Communicate system architectural design_optional

Fig. 4. Activity diagram of the System design phase

Fig. 4 shows four tasks from the plug-in *Partial commonality* (upper four) whereupon the first task has been replaced by a company/consortium-specific task (this task represents the interpretation and customization carried out in the context of the SafeCer project) and two tasks from the plug-in *Optional*. The two tasks at the bottom (named *optional*) derive from the inclusion of ASPICE-specific elements.

In the EPF-Composer a *Method Configuration* is used to define a subset of the library. This subset, which is the basis for exporting to XML and HTML, defines the packages and plug-ins that are added or subtracted. EPF-Composer permits users to export libraries, plug-ins and configurations in XML format for further processing in other process-related-tools (Fig.1.B.2). Fig. 5 shows how the subset is arranged by selecting *Plug-ins* and *Content Packages*. An element appears in the configuration if the related check box is marked otherwise it is not part of the configuration.



- ■ Base
  - ■ Phase_System_Design
    - ■ Base_System_Design_Commonality_Full
      - ☑ Method Content
        - ☑ Content Packages
          - ☑ System_Design_Base
            - ☑ [Elements]
            - ☑ System_Design_Commonality_Full
              - ☑ [Elements]
      - □ Processes
      - ☑ Categories
    - ☑ Base_System_Design_Commonality_Partial
    - ☑ Base_System_Design_Optional
    - ☑ System_Design_Variants
- ■ Processes
  - □ ASPICE_ISO26262_IEC61508_System_Design_Delivey Process
  - ☑ ASPICE_ISO26262_System_Design_Delivery Process
    - ☑ Method Content
      - ☑ Content Packages
    - ☑ Processes
    - ☑ Categories
  - □ ASPICE_System_Design_Delivery Process

Fig. 5. Selection of reusable process elements

The snippets below show how the task *Define system architectural design_V_SafeCer* in the variants plug-in is related the task *Define system architectural design* in the partial commonality plug-in. The linkage between single elements is implemented by using unique ID's.

```
<ContentElement xsi:type="uma:Task"
name="Define system architectural design"
id="_xJ8cgITREeOXwJvr4znqwg"
</ContentElement>
<ContentElement xsi:type="uma:Task"
name="Define system architectural design_V_SafeCer"
variabilityBasedOnElement="_xJ8cgITREeOXwJvr4znqwg"
variabilityType="replaces">...
</ContentElement>
```

IV. Lessons Learned

In this section we present the lessons learned that we have derived by implementing the safety-oriented process line approach in SPEM2.0/EPF. The lessons concern the following four main bolded aspects: **General soundness** - The adoption of the safety-oriented process line approach is sound since prescriptive processes mandated by the standards exhibit commonalities. The adoption is also beneficial since it enables systematic reuse of process elements. **Traceability** might be precondition for the acceptance of a process in a company. The clear relationship between the derived process and the original standard provides support for arguing process-compliance. Every user of the process is able to understand which section of a standard is base for the definition of a derived process element. For this reason a direct link to the standard is part of every process element. **Modelling limitations** - As pointed out by predecessors and as also found out through this work, SPEM2.0/EPF Composer offers a limited variability modelling support, which makes it not ideal for modelling a process line. **Flexible use of the notion of partial commonality** resulted to be a strategic solution for increasing the identification of common process elements.

V. Related work

The necessity to react to the introduction of the new ISO standard for functional safety has motivated several research works aimed at first of analysing the existing gap with respect to previous ways of working and then proposing solutions. In [19], for instance authors propose to simply extend ASPICE with safety-related process elements in order to fulfil the requirements of ISO 26262. This trend of extending/upgrading/amplifying a standard with safety-specific process elements in order to provide ad-hoc solutions, is also pioneered in [20-21]. Our approach profoundly differs from these ones. We do not pursue ad-hoc and "hard modelled" or (simply "hard written" in case of natural language-based proposals) solutions. Instead, we propose to thoroughly and systematically engineer a modelling framework supporting flexible process models definition and thus allowing process engineers to select and compose process elements in compliance with the required standard(s).

VI. Conclusion and future work

In this paper we have proposed a methodological framework for implementing the safety-oriented process line approach. More specifically, we have examined three standards that are used in the automotive domain and after having identified commonalities and variabilities we have shown how to systematically model them in SPEM2.0/EPF. Then, we have also shown how those commonalities and variabilities can be exploited for the definition of flexible processes. From this work we have drawn some lessons learned: the examined processes exhibit commonalities and thus the safety-oriented process line approach represents a sound and effective way for systematizing reuse and enabling the introduction of the changes that might be required when switching from one standard to another (e.g. for intra-domain re-certification). The current modelling support is however too limited.

In the future, in close cooperation with experienced safety assessors, we plan to extend this work by considering additional process elements and other safety related standards. The extension will include the modelling of the specific roles to synthesize the required skills to be standard compliant and ensure that a check in terms of competences has been done. Additionally, we intend to model also work products, guidance and tools. We also intend to actively contribute to the provision of a more adequate modelling support for safety-oriented process lines. Further, we aim on the definition of metrics that allow process engineers to evaluate the reduction in terms of time and cost enabled by the systematization of reuse. To be aligned with the ongoing evolution of the functional safety standards proposed by the rather influent set of automotive Swedish manufactures, we intend to consider SS-7740 [15] as well as CMMI-DEV [16] plus its corresponding extension for safety (+SAFE [20]).

Acknowledgment

This work is supported by the nSafeCer project [7].

References

[1] Automotive SPICE Process Assessment Model, Automotive SIG, 2010, www.automotivespice.com

[2] IEC61508:2010. Functional safety of electrical/electronic/programmable electronic safety-related systems.

[3] ISO26262. Road vehicles – Functional safety. International Standard, November 2011.

[4] T. Ternite. Process Lines: A Product Line Approach Designed for Process Model Development. Software Engineering and Advanced Applications, Euromicro Conference, pp. 173-180, 2009 35th Euromicro Conference on Software Engineering and Advanced Applications, 2009.

[5] B. Gallina, I. Sljivo, O. Jaradat. Towards a Safety-oriented Process Line for Enabling Reuse in Safety Critical Systems Development and Certification. Post-proceedings of the 35th IEEE Software Engineering Workshop (SEW-35), 2012.

[6] ARTEMIS-JU- 269265 pSafeCer - pSafety Certification of Software-Intensive Systems with Reusable Components.

[7] ARTEMIS-JU- 295373 nSafeCer - nSafety Certification of Software-Intensive Systems with Reusable Components.

[8] OMG. Software & systems Process Engineering Meta-model (SPEM), v 2.0. Full Specification, Object Management Group, 2008.

[9] Eclipse Process Framework, www.eclipse.org/epf/.

[10] K. Zamli and P. Lee. Taxonomy of Process Modeling Languages. Proc. ACS/IEEE Intl. Conf. Computer Sys. and Appl., Beirut, Lebanon, pp. 435–437, June 2001.

[11] S. T. Acuña, X. Ferré. Software Process Modelling. Proc. World Multiconf. Systemics, Cybernetics, and Informatics, Orlando, FL, pp. 237–242, July 2001.

[12] R. Bendraou, J.-M. Jezequel, M.-P. Gervais, and X. Blanc. A Comparison of Six UML-Based Languages for Software Process Modeling. IEEE Trans. Softw. Eng. 36, 5, pp. 662-675, September 2010.

[13] I. Ruiz-Rube, J. M. Dodero, M. Palomo-Duarte, M. Ruiz, D. Gawn. Uses and Applications of SPEM Process Models. A Systematic Mapping Study. J. Softw. Maint. Evol.: Res. Pract.; 00:1–32, Published online in Wiley InterScience. DOI: 10.1002/smr, 2012.

[14] T. Martínez-Ruiz, F. García, M. Piattini, J. Münch. Modeling Software Process Variability: An Empirical Study. IET Software, vol. 5, no. 2, pp. 172-187, 2011.

[15] SS-7740. Road vehicles - Functional Safety Process Assessment Model. SIS 2012.

[16] M. B. Chrissis, M. Konrad, S. Shrum. CMMI: Guidelines for Process Integration and Product Improvement (3rd edition) . SEI Series in Software Engineering.

[17] S. Kashiyarandi. Reusing Process Elements in Context of Safety Critical Systems Development and (re)Certification. Master's thesis, Mälardalen University, School of Innovation, Design and Engineering, Sweden (to appear in 2014).

[18] R. Bramberger, Application of process line approach for different standards in a safety-critical context. Bachelor's thesis, Graz University of Technology, Austria (to appear in 2014).

[19] P. Johannessen, Ö. Halonen, O. Örsmark. Functional Safety Extensions to Automotive SPICE According to ISO 26262. Proceedings of the 11th International Conference Software Process Improvement and Capability Determination-SPICE. Springer, pp. 52-63. Dublin, Ireland, May 30 - June 1, 2011.

[20] +SAFE V1.2, A Safety Extension to CMMI-DEV, V1.2 (CMU/SEI-2007-TN-006). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, March, 2007.

[21] E. Petry. How to Upgrade SPICE-Compliant Processes for Functional Safety. 10th International Conference Software Process Improvement and Capability Determination-SPICE. Pisa, Italy, May 18-20, 2010.

Model-based Engineering Workflow for Automotive Safety Concepts

2015-01-0273

Published 04/14/2015

Helmut Martin, Martin Krammer, Bernhard Winkler, and Christian Schwarzl
Virtual Vehicle Research Center

CITATION: Martin, H., Krammer, M., Winkler, B., and Schwarzl, C., "Model-based Engineering Workflow for Automotive Safety Concepts," SAE Technical Paper 2015-01-0273, 2015, doi:10.4271/2015-01-0273.

Abstract

Although the ISO 26262 provides requirements and recommendations for an automotive functional safety lifecycle, practical guidance on how to handle these safety activities and safety artifacts is still lacking.

This paper provides an overview of a semi-formal safety engineering approach based on SysML for specifying the relevant safety artifacts in the concept phase. Using specific diagram types, different views of the available data can be provided that reflects the specific needs of the stakeholders involved. One objective of this work is to improve the common understanding of the relevant safety aspects during the system design.

The approach, which is demonstrated here from the perspective of a Tier1 supplier for an automotive battery system, covers different breakdown levels of a vehicle.

The safety workflow presented here supports engineers' efforts to meet the safety standard ISO 26262 in a systematic way. Furthermore, it offers a solution to deal with the interaction of safety artifacts and the safety analysis activities, which allows for the creation of a compelling safety argument in the concept phase.

Introduction and Motivation

The development of cyber-physical systems - in particular in the automotive domain - has to overcome several challenges, including increasing complexity due to the inter-connection and inter-communication of distributed systems in a vehicle [1] and addressing multidisciplinary automotive systems (electrical, mechanical, chemical and thermal disciplines e.g. automotive battery systems) [2]. Furthermore, all of these systems must comply with different standards and regulations, such as quality development process standards (e.g. Automotive SPICE - IEC 15504), functional safety standards (e.g. ISO 26262 [3]), product-specific standards (e.g. test specifications for battery systems ISO 12405 [4]) and others. Most of these standards are harmonized because they refer to each other and must be applied in the same engineering domain.

There is one important principle for mastering the complexity in such systems: “divide and conquer”. By following this breakdown principle, the product and process aspects can be covered. Concerning the product aspect, a system can be hierarchically structured and separated into elements with clearly defined or even standardized interfaces. Furthermore, the engineering process can be divided into a number of more finely grained engineering processes, complemented by certain activities to support the integration of the various engineering artifacts.

The ISO 26262 “Road Vehicles - Functional Safety” is an automotive-domain-specific derivation of the generic industrial functional safety standard IEC 61508 [5]. Since November 2011, the ISO 26262 has been mandatory for the field of electrical and/or electronic (E/E) systems. It covers the complete safety lifecycle, including design, development, production, service processes and decommissioning. The standard provides guidance by introducing requirements and recommendations to reduce the risk of systematic development failures and to handle the complexity of E/E systems. Nevertheless, compliance with the standard presents a significant challenge for companies because it only sets requirements, but does not explicitly explain how these requirements can be implemented in an efficient way.

For all engineering activities, specific documentation is required to prove that the development of a product conforms to the standard. The ISO 26262 requires specific work products, which are related to requirements of the standard for each safety activity. However, different work products share the same safety artifact data. Thus, if any of these artifacts have to be changed, different work products are affected and have to be updated to provide consistent information. This is the main drawback of document-centric development. The introduction of semi-formal notation (e.g. SysML) improves this situation because the artifacts in a system model use a consistent, shared data source. The required work products can therefore be created and exported as the documentation output of the system modeling effort.

The modeling described in this paper is based on the international standard OMG Systems Modeling Language (SysML) [6], since this language has been successfully applied and is supported by commercial available modeling tools. A standardized, semi-formal modeling language paves the way from the document-centric to the model-centric development approach. However, it is difficult to use "raw" SysML with no supporting methodological background or restrictions to a specific subset of elements and associations between the modelling elements.

This work makes three contributions: It presents a safety workflow with respect to ISO 26262, it creates an integrated Safety Analysis tool chain, and if proposes a SysML profile that supports the safety workflow.

The presented workflow is a tool-independent approach based on SysML. One constraint for the development of the approach was the use of company-specific tools that are used in the existing, established engineering process, without the purchase of additional commercial tools.

Overview of ISO 26262 Safety Lifecycle

The ISO 26262 defines a safety lifecycle by delineating different levels of safety requirements (see Figure 1). The safety issues are thereby outlined from a higher level of functional abstraction (safety goal, functional safety requirements) to the more detailed levels of the technical realization of a system (technical safety requirements), down to the software (software safety requirements) and hardware levels (hardware safety requirements). The following paragraphs explain our conclusions to create new concepts for the separation between problem and solution space in the context of ISO 26262.

Functional Needs

The functional needs cover the definition of the item, which represents a system or an array of systems that realizes a specific functionality on the vehicle level ([3]-Part3).

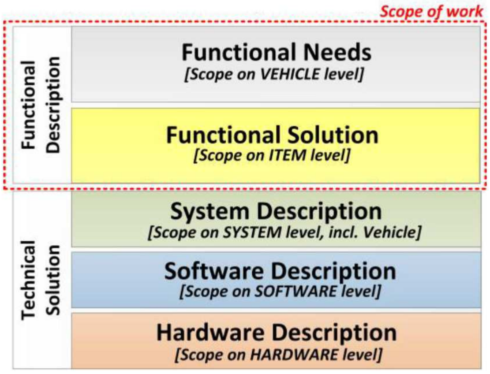


Figure 1. Overview of abstraction levels

Moreover, the preliminary architecture assumption (PAA) of the vehicle has to be elaborated, which defines the boundary of the item and the interaction of the item with different stakeholders (e.g. other items, other technologies, users/ humans). Based on this "item definition", the Hazard Analysis and Risk Assessment (HARA) analyze all potential risks from hazardous events. The HARA determines an Automotive Safety Integrity Level (ASIL) for the risk classification and a number of safety goals. The safety goals represent the top-level safety requirements for the subsequent safety activities in the safety lifecycle and define additional attributes for the vehicle (e.g. safe state). The ASILs and the safety goals correspond to the hazardous events.

The technical solutions are derived from functional solutions. The abstract functional solution of the item has to be defined independently of any technical solution, so that it can be reused for different kinds of implementations. Furthermore, this separation is crucial for the understanding of the characteristics and interactions of today's multidisciplinary automotive systems.

Functional Solution

The first level of the solution is covered in the functional safety concept (FSC) ([3]-Part 3), which has to be derived from the safety goals. The FSC defines all required safety measures for handling different kinds of identified malfunctions. These safety measures have to be reflected as functional safety requirements (FSR), which must be allocated to the elements of the PAA of the item.

In this paper, we focus on the functional description described above, which covers the functional needs and functional solution. It is a very important part of the safety lifecycle because the ASIL and the defined functional safety requirements provide the basis for the technical solution and all the subsequent safety activities.

Technical Solution

In the following technical solution, the description of the system, software and hardware defines the technical implementation.

The system description ([3]-Part 4) defines the technical safety requirements (TSR), which refine the FSC by specifying safety mechanisms. Safety mechanisms, which are technical measures related to the detection, indication and control of faults both in the system and in external devices that interact with the system, define measures to achieve the defined safe state.

The system design specification and the technical safety concept have to comply with the FSR and TSR of the item. The allocation of TSR to system design elements has to be performed, and the system design has to implement these TSR, which is documented in the technical safety concept (TSC). Safety analyses of the system design are needed to avoid systematic failures, random HW failures, and common cause failures, as well as to verify the TSC.

On the lowest level of the system implementation, the TSR have to be refined and allocated to the hardware and software components. The hardware/software interface has to be specified on the system integration level for the coordination between component hardware devices that are controlled by software and hardware resources that support the execution of software.

The software description ([3]-Part 6) covers all safety activities concerning the software development. The software-relevant supporting processes are initiated by identifying the appropriate methods in order to comply with the requirements and their respective ASIL.

The hardware description ([3]-Part 5) covers the activities and processes necessary for the product development at the hardware level, including the hardware implementation of the technical safety concept, the analysis of potential hardware faults and their effects, and the coordination with software development.

The following section summarizes and discusses the relevant standards and related literature, after which the "Methodology" describes the presented approach for the safety workflow, including the system modeling and safety analysis. The usability of the approach is then demonstrated using the case of an automotive battery system, and the results are described in the "Demonstration" section. In the "Discussion" section, we provide some lessons learned from the application of the approach, and the "Conclusion and Outlook" section concludes the paper and provides ideas for further investigations.

Related Work

The work of Dajusuren, et al. [7] discussed a number of Architecture Description Languages (ADLs) (SysML, EAST-ADL, AADL, TADL, AML, and MARTE) and evaluated this set of ADLs based on the automotive-specific modeling requirements. They selected SysML as a viable language to carry out the case study on automotive systems and to demonstrate a method for architectural consistency checking using SysML. The use of the SysML diagram types was evaluated, and the benefits and disadvantages of the features were discussed from the perspective of the automotive domain.

Papadopoulos, et al. described a method [22] and a tool [23] for the automatic generation of an FMEA. The tool constructs FMEAs from engineering diagrams that have been augmented with information about component failures.

Piques, et al. described their industrial experiences in applying a SysCARS methodology in industrial projects [8]/[9]. The SysCARS methodology provides a precise mapping of system engineering artifacts to SysML artifacts, as well as the sequence of modeling of the activities to be performed, by a "workflow-driven" mechanism. Moreover, they show how interoperability is ensured with the tools already in place for requirements management and control design.

Martin, et al. provided a state-of-practice study about the reusability of safety artifacts [10]. Their work identified the need for Model Based Systems Engineering (MBSE) to improve the state of industrial practice. MBSE provided the fundamental basis for applying any kind of reuse approaches because any reusable artifact can be handled and managed systematically, in contrast to the document-centric approach, where potentially reusable data is hidden in text fragments in natural language.

The present work used three safety analysis methods: HAZard and OPerability study (HAZOP), Hazard Analysis and Risk Assessment (HARA), and Failure Mode and Effects Analysis (FMEA).

The HAZOP analysis is a team-based method for identifying potential safety and operational problems associated with the design, maintenance or operation of a system. A HAZOP is a formal and objective process, which ensures a systematic and well-documented evaluation of potential problems/hazards [11][12].

The HARA is a safety analysis workflow for the automotive domain which provides a systematic determination of the potential risks in specific driving situations, introduces the Automotive Safety Integrity Level (ASIL) and defines Safety Goals ([3]-Part 3).

The FMEA is a systematic method of analyzing potential failure modes aimed at preventing failures. It is intended to be a preventive action process carried out before implementing new features or changes in products or processes [13][14].

The functional safety standard ISO 26262 defines the Safety Case as follows: "*The Safety Case argument that the safety requirements for an item are complete and satisfied by evidence compiled from work products of the safety activities during development*." ([3]-Part 1). The Safety Case can be interpreted as complete safety justification, including all the supporting material, which provides evidence such as relevant design information, verification and validation reports. The Safety Case Report summarizes and references all the supporting documentation in a clear and concise format. The University of York has been investigating the topic of Safety Cases [15] for many years, in particular for the avionic and automotive domains. The research group around Tim Kelly introduced a semi-formal modeling language called Goal Structuring Notation (GSN)[16][17]. GSN is a graphical argumentation notation that can be used to document explicitly the individual elements of any argument (claims, evidence and contextual information) and, perhaps more significantly, the relationships that exist between these elements. Arguments documented using GSN can help provide assurance of critical properties of systems, services and organizations (such as safety or security properties). The present paper uses the GSN modeling language for the semi-formal notation of the preliminary safety case. The University of York organized a GSN Working Group, which released the GSN standard[1] in November 2011 to provide guidance on the usage of the GSN modeling language.

Methodology

The ISO 26262 prescribes a safety lifecycle for the development of E/E systems which covers all relevant phases with specific requirements for the different safety activities. These safety activities can be categorized into six different aspects (see Figure 2).

Requirements [REQ]: Elements for requirements (REQ) definition, which includes REQ ID, REQ text, REQ attributes

Architecture [ARCH]: Static architectural definitions that define the ARCH element and its interfaces.

Behavior [BEH]: Functional descriptions (static and dynamic)

Safety Analysis [SAF]: Covers all kinds of elements that are needed to perform safety analysis (e.g. failure model)

[1] http://www.goalstructuringnotation.info/

Verification Validation [V&V]: Covers all V&V criteria and test cases that provide evidence and approve the specification

Safety Argumentation [ARG]: Covers all kinds of elements that are needed to create the safety case

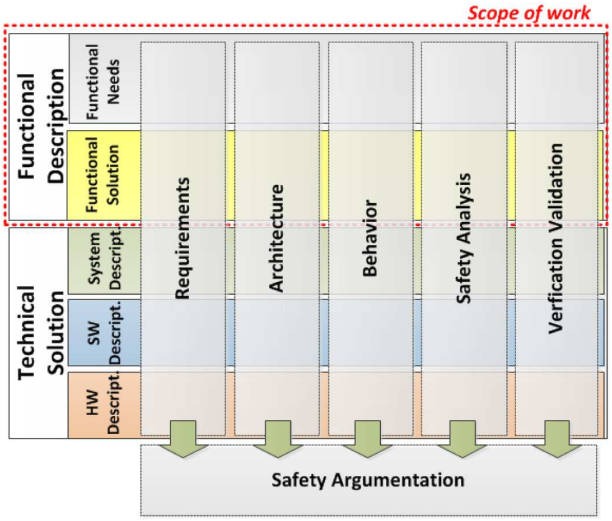


Figure 2. Overview of safety artifact categories

To provide evidence of the adequate fulfillment of all these requirements, work products have to be elaborated. These work products integrate the documentation of the safety artifacts (e.g. the item definition contains preliminary architecture and functional description).

Introduction of Integration Levels

To cover the different levels of integration in a vehicle, we defined a structure that covers these aspects by outlining these levels in a breakdown of a Hybrid Electric Vehicle (HEV) (see Figure 2):

- **Level 0** - Vehicle Level (e.g. Hybrid Electric Vehicle)
- **Level 1** - Vehicle Components (e.g. Hybrid Powertrain (HPT), Chassis,…) [Level 0 - Components]
- **Level 2** - HPT Components (e.g. HCU, Battery System) [Level 1 - Components]

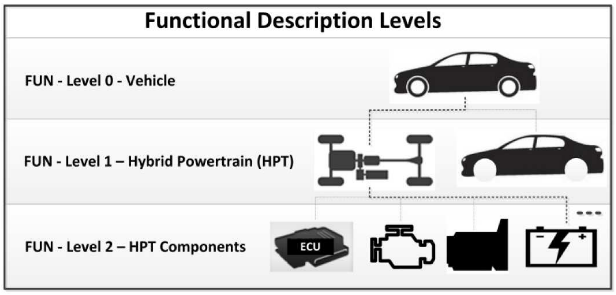

Functional Description Levels
FUN - Level 0 - Vehicle
FUN - Level 1 – Hybrid Powertrain (HPT)
FUN - Level 2 – HPT Components
ECU

Figure 3. Overview of integration levels (e.g. hybrid electric vehicle)

Safety Workflow to Develop System Model

This section provides an overview of the main safety activities of the functional abstraction in the concept phase of ISO 26262:

1.). **Activity 1: Analyzing Functional Needs**
2.). **Activity 2: Defining Functional Safety Concept**

Activity 1 ("Analyzing of Functional Needs") starts with the definition of the item, which defines the intended functionality of the item and provides an assumption of the preliminary architecture of the vehicle, the boundary of the item and the interaction with other vehicle items or technologies and with different kinds of users (humans). Based on this item, the HARA is defined. The HAZOP safety analysis method supports the hazard identification. During the HARA, the potential risks (i.e. hazards) have to be identified and combined in specific driving situation as "hazardous events". The final task is determining the ASIL and defining the safety goal.

- **Activity 1: Analyzing Functional Needs:**
  - Task 1.1: Item Definition
  - Task 1.2: Hazard Analysis and Risk Assessment (HARA)
  - Task 1.3: ASIL Determination + Safety Goal Definition
  - Task 1.4: Argumentation of Safety for Problem Description

Activity 2 -"Definition of Functional Safety Concept" has to provide an initial abstract functional solution documented in the functional safety concept (FSC) ([3]-Part 3). The FSC defines all required safety measures, which represent functional solutions to fault detection and failure mitigation, fault tolerance mechanisms, the transition to safe state, possibilities for driver warnings, and other aspects. These safety measures have to be expressed as functional safety requirements, which have to be elaborated in the context of the safety goal defined at level 0. All relevant safety measures for the specific item have to be clarified, as well as all relevant external measures or other technologies outside of the item which are taken into account for the item's FSC. Safety analysis methods such as FMEA can be used to support the identification of the potential risk. The Concept FMEA provides an initial verification of the functional safety concept and the defined safety measures for the prevention and detection of identified failure modes. Finally, the identified functional safety requirements (safety measures) have to be allocated to the assumed preliminary architectural elements of the item, which forms the basis for the subsequent technical solutions.

- **Activity 2: Definition of Functional Safety Concept**
  - Task 2.1: Derivation of Functional Safety Requirements
  - Task 2.2: Safety Analysis (Concept FMEA)
  - Task 2.3: Allocation of Functional Safety Requirements
  - Task 2.4: Definition of Validation Criteria
  - Task 2.5: Argumentation of Safety for Functional Solution

Task Definition

Once the activities have been defined and the tasks have been derived, more details must be elaborated for each 'Task' to support the practical execution of the safety workflow. The following information, represented by specific Work Products (WP) and/or artifacts, is required as inputs or outputs for the performance of specific steps for each task:

- *Input*: WP and/or relevant artifacts

- *Steps/Questions*: Specific steps or questions support the definition of the required artifacts, depending on the task.
- *Output:* WP and/or relevant artifacts

The following section provides an exemplary definition of a task of *Activity 1: Analyzing of Functional Needs* and describes the modeling artifacts needed to model this task.

@Task 1.1: Item Definition:

- *Inputs:* All relevant inputs for defining the item have to be collected and documented.
- *Steps/Questions:*
  - What does a preliminary architecture look like?
  - Which use cases of the vehicle are important?
  - Which stakeholders are involved (persons, external systems)?
  - Which functions of the item are relevant?
  - Which operating modes and states of the item have to be covered?
  - What failure modes and hazards are known?
  - …
- *Outputs:*
  - WP: Item Definition
  - Artifacts:

    [ARCH]: Preliminary Architecture Elements, Boundary, Interfaces

    [BEH]: Use Cases of the vehicle incl. different stakeholders of the item; Abstract (Item) Functions; Operating modes and states of the item

    [SAF]: Known failure modes and hazards

Interaction of System Model and Safety Analyses

On all these different levels of integration, the system model and the different safety analysis methods must interact in order to identify all relevant malfunctions.

Figure 4 provides an overview of the safety analysis method used in this work, as well as which level of integration is involved (notice the numbering in the circles in the figure):

1.). HAZOP: Level 1 and Level 0 (yellow bar)
2.). HARA: Level 0 (orange bar)
3.). Concept FMEA: Level 0, Level 1, and Level 2 (green bar), where Level 1 is the central level of the safety analysis

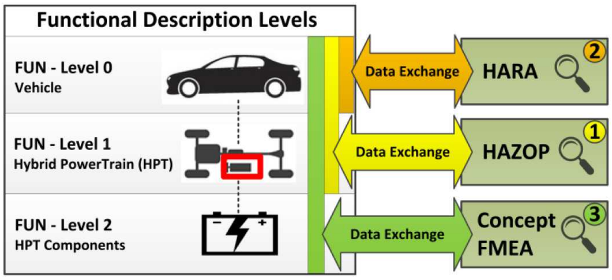


Figure 4. Interaction of system model and safety analyses

Furthermore, the different safety analysis methods support the definition of adequate safety measures corresponding to specific malfunctions of the item.

Interaction System Model ⇔ HAZOP

In part 3, ISO 26262 states: "7.4.2.2.1 -The hazards shall be determined systematically by using adequate techniques".

Based on this requirement, the HAZOP is used in our approach to identify potential vehicle hazards as a basis for the HARA. The HAZOP analysis procedures involve taking the available description of a system and systematically questioning every part of it to establish the possible deviations that might arise due to an unintended functionality of the system. In the next step, the consequences of those deviations are assessed by defining the potential hazards that could have a negative effect on the vehicle. The HAZOP is applied in a structured way by a team of domain experts (e.g. Systems-/ HW-/ SW-/ mechanical-engineers), who are responsible for the implementation of the system in later development phases.

Note: In this approach, the focus of the HAZOP is on hazard identification. No further recommendations or requirements of a standard HAZOP are required.

A complete HAZOP requires the following information:

1. Item under analysis
2. Guidewords
3. System effect if guideword occurs
4. Resulting hazard or deviation

Table 1. Input/Output artifacts of HAZOP

Input		Output	
Type	Import Artifacts	Type	Export Artifacts
ARCH	Component Element	SAF	Identified Hazards (associated with component and function)
BEH	Abstract Functions		

Interaction of System Model ⇔ HARA

The HARA is an analysis workflow defined in the ISO 26262 - Part 3, which has to be documented in the work product of the same name. It is performed by a team, which should involve not only experts from the different domains, but also some kind of user perspective, which could be provided by managers or administrative staff in a company. In the presented workflow, the HARA uses the output of the HAZOP as a basis for identifying hazards and defining hazardous events. For the hazardous events, it is necessary to analyze potential situations and combine them with the identified hazards. The team must evaluate these hazardous events using three independent risk assessment parameters: severity (S from 0 to 3), exposure (E from 0 to 4), and controllability (C from 0 to 3). For each classification of these parameters, a rationale is needed to document all assumptions. If any of these parameters can be classified as "0", then there is no safety criticality of the hazardous event. However, even in such cases, the rationale for that decision must be provided.

The Automotive Safety Integrity Level (ASIL) can be determined by combining these three parameters in accordance with the risk table in the ISO 26262. The ASIL covers a range of QM ASIL A/B/C/D, where QM means Quality Management (QM) and no safety criticality, ASIL A is the lowest integrity level, and ASIL D is the highest integrity level. The ASIL is a vital parameter for all subsequent safety activities of the safety lifecycle because it corresponds to the number of safety requirements and recommended methods. Thus, the ASIL can be seen as an indicator of the engineering effort required to provide evidence of the fulfillment of ISO 26262.

The following steps must be covered in the HARA on the vehicle level:

1. Hazard identification (done by HAZOP)
2. Situation analysis
3. Classification of hazardous events (S/E/C)
4. Determination of ASIL and safety goals
5. Verification of the HARA

Table 2. Input/Output artifacts of HARA

Input		Output	
Type	Import Artifacts	Type	Export Artifacts
SAF	Identified Hazards	REQ	Safety Goals + ASIL X (associated with the identified Hazards)

Interaction of System Model ⇔ Concept FMEA

The concept FMEA is a crucial method in the concept phase for (1) the identification of possible hazards for the later hazard analysis and risk assessment and (2) the derivation of E/E and non-E/E safety measures for the functional safety concept.

The following discussion shows the interaction between the system model and the Concept FMEA on the functional level and analyzes "what the system does", without covering "how the system does it". The Concept FMEA has to be performed by a team of domain experts who know their relevant system components and provide input about their failure behavior.

Using a functional abstraction, the functions have to be analyzed based on the following failure modes ([11] - Ch. 13.5.3):

- Function fails to perform
- Function performs incorrectly
- Function performs prematurely
- Function provides incorrect information
- Function does not fail safe

To perform the Concept FMEA, the following artifacts have to be provided as input: structural elements as a structure net, functions as a function net (associated with structure elements), and malfunctions in a failure net (associated with functions) from vehicle level down to the relevant level of the item (see Figure 5). The structure net describes relations between system components, the function net describes the relations between functions, and the failure net describes the relations between failures. Each structure element relates to one or more functions, and each function could relate to one or more malfunctions.

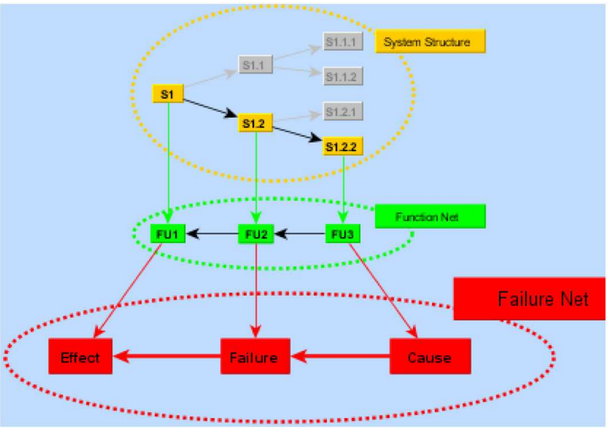


Figure 5. Relations between system structure, functional net, and failure net (Source: APIS IQ[2]).

The following steps must be taken to perform the Concept FMEA:

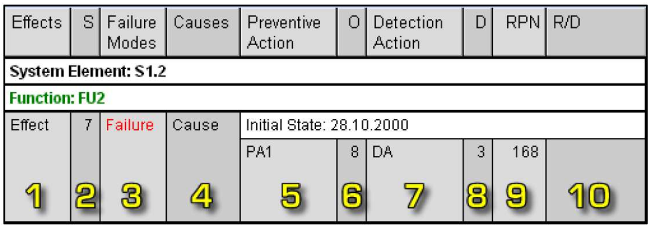
1. Conduct FMEA
   a. *Define the system components and create the structure net, including all system components*
   b. *Define the functions of all system component and create the function net of all functions*
   c. *Identify all malfunctions*
2. Evaluate malfunctions by Risk Priority Number (RPN)
3. Recommend/Define actions for
   a. *Corrective/Preventive actions*
   b. *Monitoring/Detection actions*
4. Re-evaluation of malfunctions by RPN with respect to recommended Actions (3+4) that cover a defined RPN limit

Table 3. Input/Output artifacts of Concept FMEA

Input		Output	
Type	Import Artifacts	Type	Export Artifacts
ARCH	Architectural Elements (Structure Net)	REQ	Preventive Actions (Corresponding to Malfunction)
BEH	Abstract Functions (Function Net) (Corresponding to Functions)	REQ	Detection Actions (Corresponding to Malfunction)
SAF	Malfunctions (Failure Net)		

[2] FMEA tool APIS IQ FMEA - http://www.apis.de/en

To perform the Concept FMEA, we use a worksheet that covers the safety-related information, as well as reliability information. Figure 6 shows a template for the Concept FMEA.

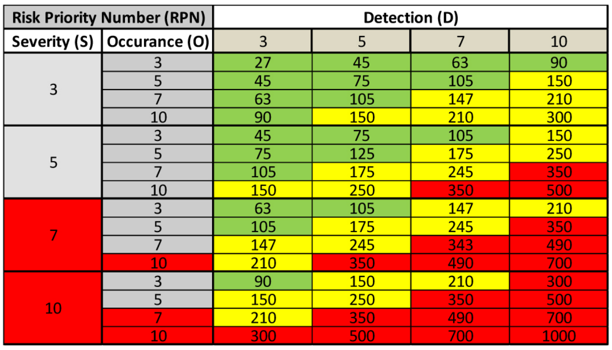


Effects	S	Failure Modes	Causes	Preventive Action	O	Detection Action	D	RPN	R/D
System Element: S1.2									
Function: FU2									
Effect	7	Failure	Cause	Initial State: 28.10.2000					
				PA1	8	DA	3	168	
1	2	3	4	5	6	7	8	9	10

Figure 6. Template of worksheet for Concept FMEA (Source: APIS IQ)

The RPN is used to express the risk of a system in terms of reliability. The expert team must determine three parameters in the FMEA: severity (S from 1 to 10), occurrence (O from 1 to 10), and detection (D from 1 to 10). The product of these individual ratings in the FMEA results in the Risk Priority Number [RPN = S*O*D]. For the Concept FMEA, we decided to use a rough scaling by reducing the scales to the values {3,5,7,10} because the fine scale is too detailed for this initial analysis. To identify a specific failure-mode that has to be treated as safety critical, we defined a threshold for the resulting RPN > 300 (marked in red), as shown in Figure 7.

After the initial determination of the RPN, no special preventive or detection actions are defined. In the next step, Preventive Action (PA) and Detection Action (DA) must be defined and the new corresponding RPN must then be determined. These actions directly influence the parameters of occurrence (PA ⇒ O) and detection (DA ⇒ D). After defining these actions, a further iteration of the RPN parameter determination must be performed.



Risk Priority Number (RPN)		Detection (D)			
Severity (S)	Occurance (O)	3	5	7	10
3	3	27	45	63	90
	5	45	75	105	150
	7	63	105	147	210
	10	90	150	210	300
5	3	45	75	105	150
	5	75	125	175	250
	7	105	175	245	350
	10	150	250	350	500
7	3	63	105	147	210
	5	105	175	245	350
	7	147	245	343	490
	10	210	350	490	700
10	3	90	150	210	300
	5	150	250	350	500
	7	210	350	490	700
	10	300	500	700	1000

Figure 7. RPN matrix incl. the highlighting of relevant RPN values (red… safety relevant, yellow…reliability relevant)

From a safety-critical perspective, the influence of the actions has to influence the rating of the resulting RPN < 300. (From a reliability perspective, the limit was defined as RPN < 145, which is indicated in yellow.)

Interaction of FMEA and Function Safety Concept

The Preventive Actions (PA) and Detection Actions (DA) of the Concept FMEA will be used as inputs for the verification and extension of the definition of safety measures with functional safety requirements. Since the PA and DA are linked to specific system malfunctions, each existing malfunction has to be linked to Functional Safety Requirements, if it contributes to a specific safety-critical hazardous event with an ASIL rating.

System Modeling of Safety Aspects in SysML

The major goal of a system model is to provide a single-source artifact repository, which covers all kind of artifacts that are relevant for specifying the system in a system model. This system model provides the basis for all kinds of analyses that have to be performed from different viewpoints (e.g. functional safety analysis activities). For the safety activities, some specific system modeling artifacts are needed as inputs. After the safety activities, such as safety analysis, some of these artifacts have to be extended or updated with safety-specific attributes.

The system model should cover the following aspects:

- Definition of ARCH-blocks
- Connection Interfaces of ARCH-blocks (information, physical flows)
- Allocation of functions to ARCH blocks
- Allocation of malfunctions to functions
- Allocation of hazards to malfunctions
- Allocation of FSR to ARCH-blocks

Using SysML, we introduced the following elements for the functional architecture model:

- Functional Architecture
- Connection
- Functional Element
- Function
- Malfunction
- Hazard

These elements are related as follows (see Figure 8):

- Functional Architecture relates to Functional Elements
- Functional Architecture relates to Connections
- Functional Element relates to Functional Sub-Elements
- Functional Elements relate to Functions
- Functional Elements provide Connections:

a. Requester: Functional Input

b. Provider: Functional Output

- Functions consists of Sub-Functions
- Functions relate to Malfunctions
- Malfunctions relate to Hazard (on top level)

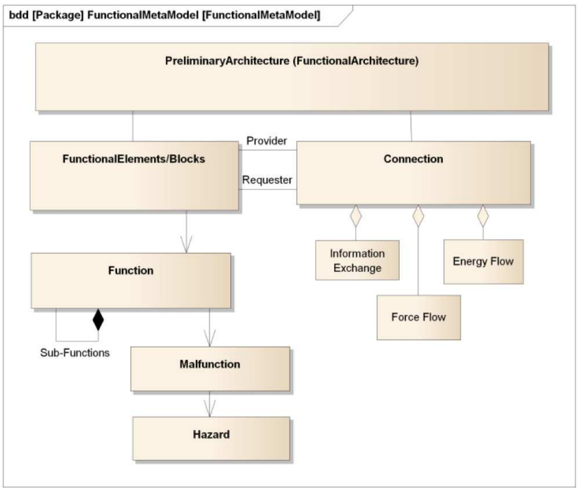

bdd [Package] FunctionalMetaModel [FunctionalMetaModel]
PreliminaryArchitecture (FunctionalArchitecture)
FunctionalElements/Blocks
Provider
Requester
Connection
Information Exchange
Energy Flow
Force Flow
Function
Sub-Functions
Malfunction
Hazard

Figure 8. Overview of functional architecture modeling elements

Data Exchange between SysML and FMEA

To exchange data between the SysML model and the FMEA, we defined the following steps (see Figure 9):

1. Create SysML model
2. Transform SysML model to FMEA model (Connection Layer)
3. Perform FMEA
4. Export results from FMEA model into SysML model (Connection Layer)
5. Allocate FMEA results to SysML model

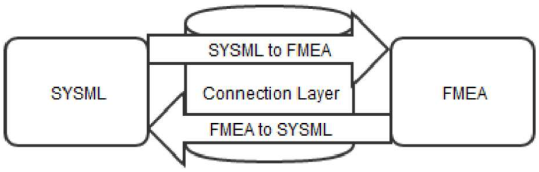


Figure 9. Data exchange between SysML and FMEA

This data exchange can only be executed with a defined tool chain.

Safety Argumentation by Safety Case

The ISO standard addresses three main elements, requirements (e.g. safety goals), argument, and evidence (e.g. work products), where the safety argument communicates the relationship between the evidence and the objectives/requirements ([3]] - Part 10).

The ISO requires more than 100 such work products as evidence for the fulfillment of the requirements for safety activities within the safety lifecycle. The problem lies in providing several pages of evidence without any clear explanation about how this evidence relates to the safety requirements. Thus, both the argument and the evidence are crucial elements for the safety case. In our approach, we decided to use Goal Structuring Notation (GSN) to represent the individual elements of a safety argument.

The creation of the safety case can be seen as an incremental activity that must be performed in parallel with engineering activities of the safety lifecycle, and incremental versions of the safety case report should be available. The preliminary safety case has to be ready after the functional safety concept has been finalized.

The GSN defines the following elements: Goals, Strategies, Solutions, Contexts, Assumptions, and Justifications. These core elements are linked using the following types of relationships: SupportedBy and InContextOf.

The modeling of the safety case should be supported by the modeling tool Enterprise Architect (EA). A GSN tool extension was created for the modeling tool EA, which provides support for the modeling of GSN modeling elements and relationships.

The different types of safety argument structure representations can be systematically reused based on the concept of introducing Safety Case Pattern (SCP) [17]/[19]/[20]. The SCP describes a partial solution and addresses one aspect of the overall structure of the safety argumentation of a safety case. The related literature contains some relevant pattern catalogues, which have to be evaluated and instantiated for the specific needs. Figure 10 shows an exemplary pattern, and the demonstration section below shows its application. In this pattern, the implicit definition of ‘safe’ is ‘hazard avoidance’. The goal G1 is covered by argument S1 that all identified hazards have been addressed. This strategy can only be executed in the context C1 of some knowledge of the plausible hazards (e.g. as identified by HAZOP). Given this information (C1), which identifies n hazards, n sub-goals of the form G2 can be constructed. The argument then develops from these ‘hazard avoidance’ goals.

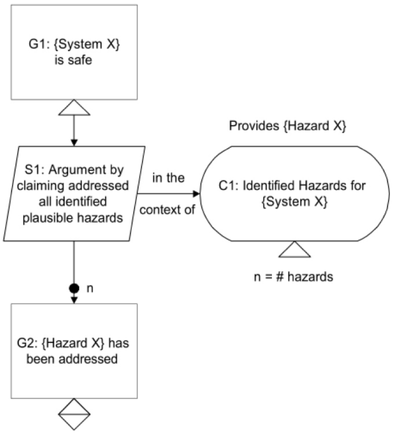

G1: {System X} is safe
Provides {Hazard X}
S1: Argument by claiming addressed all identified plausible hazards
in the context of
C1: Identified Hazards for {System X}
n = # hazards
n
G2: {Hazard X} has been addressed

Figure 10. Safety Case Pattern “All identified plausible hazards argument” described in [17]

Demonstration

This section demonstrates the usability of the approach by applying the methodology to the use case of an automotive battery system, which shows the coverage of the different aspects of safety measures.

Use Case Description

The automotive battery system is one major component of the powertrain of a Hybrid Electric Vehicle (HEV). A typical HEV powertrain consists of multiple actuators (e.g. engine, motor, clutch, transmission), which are controlled by embedded controllers, such as the Motor Control Unit (MCU), Engine Management System (EMS) or Battery Management Unit (BMU). The controllers are connected via a bus system to a coordinating master Hybrid Control Unit (HCU). Figure 1 illustrates the main components of an HEV and their electrical interconnection.

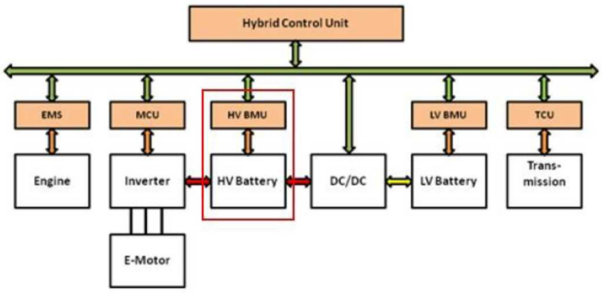


Figure 11. Hybrid powertrain: E/E architecture showing the main mechanical and electrical components, their respective control units and their electrical interconnections

The current work focuses on the battery system. The constant evolution of state-of-the-art technology is resulting in the availability of a wide variety of high-voltage battery technologies with diverse characteristics, such as lead acid, lithium-ion, lithium polymer, and nickel-metal hydride batteries. Some of the main targets for batteries to be used in HEV are low costs, high power density (e.g. 1200W/kg), very high cycle lifetime (e.g. 200,000 cycles of charge/discharge), high lifetime (e.g. 9 years), and safety.

The main functions of the battery system are:

- Provide electrical energy
- Store/charge electrical energy
- Electrical management of the battery system

Possible malfunctions are:

- Deep discharging of battery cells
- Overheating of battery cells
- Charging by deep discharged battery cells
- Overcharging of battery cells.

These malfunctions could lead to the following possible hazards:

- High voltage
- Leakage / venting gas
- Fire
- Explosion

System Modeling in Enterprise Architect

For the system model in SysML, we used Enterprise Architect (EA)[3], which is a commercial modeling tool that provides good user support and possibilities for specific tool extensions by so-called model-driven generation (MDG) technologies. The MDG technologies allow users to extend EA's modeling capabilities to specific domains and notations.

The system model covers all kinds of interactions and influences between the battery system and the vehicle behavior. To this end, the SysML model shows the connections between components, functions, vehicle malfunctions, powertrain and battery system level. Each component is represented by a functional block, and the functions and malfunctions are represented by use case elements. Figure 12 shows the connections between component, function and malfunction, which are realized via associations.

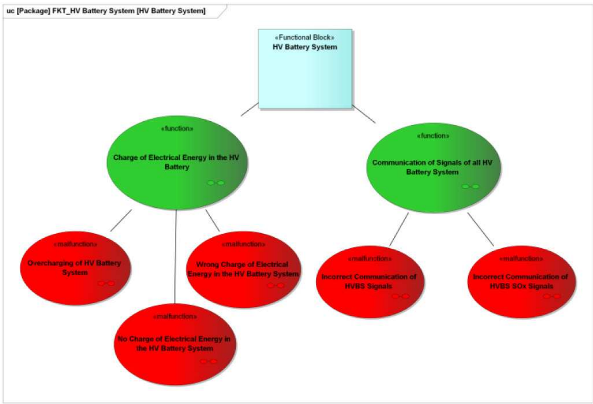

uc [Package] FKT_HV Battery System [HV Battery System]
«Functional Block»
HV Battery System
«function»
Charge of Electrical Energy in the HV Battery
«function»
Communication of Signals of all HV Battery System
«malfunction»
Overcharging of HV Battery System
«malfunction»
Wrong Charge of Electrical Energy in the HV Battery System
«malfunction»
No Charge of Electrical Energy in the HV Battery System
«malfunction»
Incorrect Communication of HVBS Signals
«malfunction»
Incorrect Communication of HVBS SOx Signals

Figure 12. Use case diagram of functions and malfunctions of the battery system

One aim of this system model is to create the three main nets - the structure net, the function net and the failure net - which are later used as inputs for the Concept FMEA.

Figure 13 shows the breakdown of the components, functions and malfunctions. The breakdown of the components is modelled by the internal block diagram, and the breakdown of the functions and malfunctions is shown in use case diagrams.

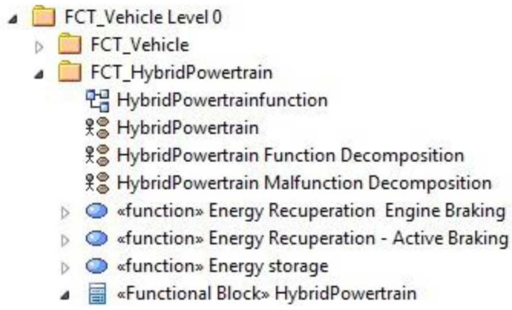

FCT_Vehicle Level 0
FCT_Vehicle
FCT_HybridPowertrain
HybridPowertrainfunction
HybridPowertrain
HybridPowertrain Function Decomposition
HybridPowertrain Malfunction Decomposition
«function» Energy Recuperation Engine Braking
«function» Energy Recuperation - Active Braking
«function» Energy storage
«Functional Block» HybridPowertrain

Figure 13. SysML structure and breakdown

[3] http://www.sparxsystems.com.au/

Based on the single source principle, every relevant safety artifact is represented in the SysML model. Each safety artifact is modeled with SysML modeling elements (e.g. the preliminary architecture) or a link to an external safety work-product (e.g. the item definition can be provided in MS WORD).

The SysML model represents the full traceability between the different safety artifacts. For the execution of the safety analysis, the required safety artifacts are transferred from EA to the external safety analysis tool. After the safety analysis, the results are transferred back to EA. Following the safety workflow, the first safety analysis is the HAZOP for identifying hazards on the vehicle level caused by malfunctions of the battery system.

For the HAZOP, a MS EXCEL template is used. The functions of the battery system are imported from EA. By combining the functions with the guidewords, the malfunctions of the battery system and malfunctioning behavior on the vehicle level are determined. From the malfunctioning behavior on the vehicle level, the possible hazards are derived. Hazards are the potential source of harm for the driver, passengers or pedestrians. The identified hazards are exported to EA and will be used as input for the HARA.

For the HARA, an additional Excel template is used. The hazards are imported from EA and combined with driving situations in an assessment matrix to derive the hazardous events. Based on the risk matrix of ISO 26262, these hazardous events are assigned a risk parameter severity (S), exposure (E), and controllability (C), and the ASIL for every hazard event is then determined. Based on the classification, the safety goals and their associated ASILs are identified. The safety goals are then exported to the EA. Based on these safety goals, the functional safety concept is defined, and the preliminary architecture is modeled in EA.

Figure 14 shows the preliminary architecture of the hybrid powertrain via the internal block diagram.

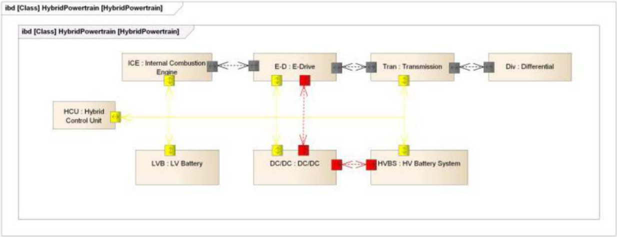

ibd [Class] HybridPowertrain [HybridPowertrain]
ibd [Class] HybridPowertrain [HybridPowertrain]
ICE : Internal Combustion Engine
E-D : E-Drive
Tran : Transmission
Div : Differential
HCU : Hybrid Control Unit
LVB : LV Battery
DC/DC : DC/DC
HVBS : HV Battery System

Figure 14. Preliminary architecture of hybrid powertrain

The FMEA is used to derive suitable safety measures to support the elaboration of the functional safety concept for the battery system. For the FMEA, the APIS IQ FMEA tool was used because it is an established tool for reliability analysis by quality engineers in the automotive industry. Furthermore, this tool can be extended to be used for functional safety analysis aspects. Thus, it is possible to combine the quality and safety analyses because the same expert team is involved in the analysis activities.

The FMEA is used to define safety measures (i.e. detection action and preventive action) for the battery system and to verify the HAZOP. The FMEA is performed in the APIS IQ tool. From EA, the modeled structure net, function net and failure net are imported. The aim of the FMEA is to define safety measures, which are called detection and preventive actions in the FMEA. The defined actions are exported to the EA as basis for the functional safety requirements.

Figure 15 shows the derivation of the preventive action to the defined FSR in a requirements diagram in EA.

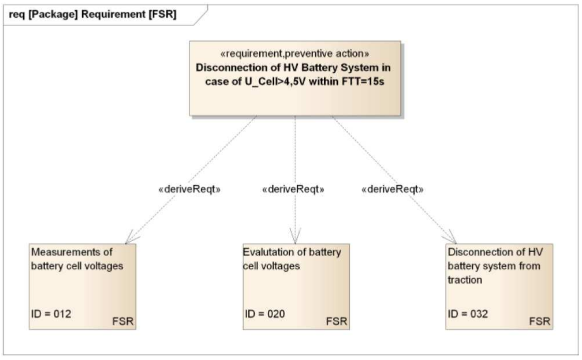

req [Package] Requirement [FSR]
«requirement,preventive action»
Disconnection of HV Battery System in case of U_Cell>4,5V within FTT=15s
«deriveReqt»
«deriveReqt»
«deriveReqt»
Measurements of battery cell voltages
ID = 012
FSR
Evalutation of battery cell voltages
ID = 020
FSR
Disconnection of HV battery system from traction
ID = 032
FSR

Figure 15. Derivation of FSR from FMEA actions

Safety Argumentation by GSN

For the safety argumentation, an MDG technology for GSN in the EA tool is used. The developed MDG technology for GSN allows the user to model a safety case for the battery system, which is consistent with the requirements and analysis results. Figure 16 shows an example for the safety case of the battery system. The advantage of the MDG technology for GSN in the EA tool is that the traceability between the GSN model and the system model can be easily shown. This helps the safety engineer and the assessor to follow the argumentation in the relevant safety case.

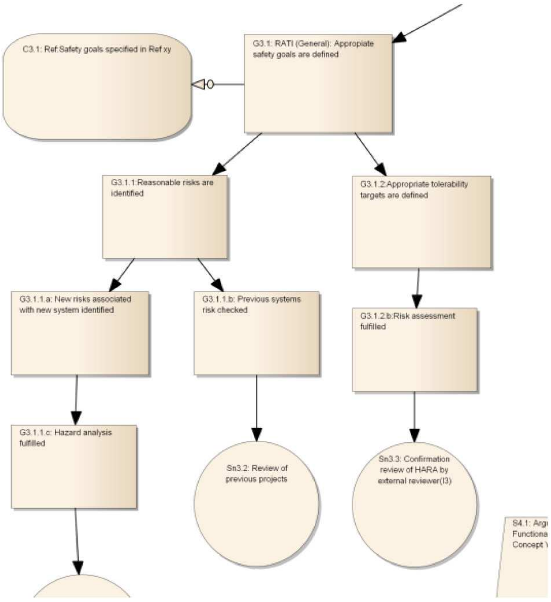

C3.1: Ref:Safety goals specified in Ref xy
G3.1: RATI (General): Appropiate safety goals are defined
G3.1.1:Reasonable risks are identified
G3.1.2:Appropriate tolerability targets are defined
G3.1.1.a: New risks associated with new system identified
G3.1.1.b: Previous systems risk checked
G3.1.2.b:Risk assessment fulfilled
G3.1.1.c: Hazard analysis fulfilled
Sn3.2: Review of previous projects
Sn3.3: Confirmation review of HARA by external reviewer(I3)
S4.1: Arg
Functiona
Concept

Figure 16. GSN model in the system modeling tool (excerpt)

Discussion

After the approach had been demonstrated, the involved industry partner evaluated the results. The achieved advantages (Adv) and potential improvements (Imp) were identified based on the lessons learned when the presented approach was applied.

Advantages of the approach

Adv#1: Common source of safety artifact data: The model-based safety engineering approach provides support for safety activities in the functional safety process as defined by ISO 26262. The approach supports a systematic description and analysis of different kinds of safety artifacts in a common system model.

Adv#2: Traceability within safety artifacts: All functional aspects are represented, including coverage of the dependencies between different kinds of safety artifacts.

Adv#3: Data exchange between model and safety analysis: Bi-directional data exchange between system modeling tool and functional safety analysis tools; Export of relevant artifacts for specific safety analysis methods and import of preventive and detection actions as a basis for deriving functional safety requirements

Adv#4: Modeling of safety argumentation: Support for creating a preliminary safety case using the GSN modeling language in accordance with the GSN standard

Potential improvements of the approach

Imp#1: Support of safety workflow: The modeling should support a safety workflow by providing guidance and templates for better usability of the SysML modeling profile.

Imp#2: Definition of complete safety modeling profile: The profile mechanism of UML/SysML should be investigated for the definition of a safety modeling profile that defines a reduced subset of modeling elements and their associations for the whole safety lifecycle.

Imp#3: Check of safety modeling artifacts: Based on the defined safety modeling profile (see Imp#2), the created artifacts of a specific safety activity could be checked to see if they are modelled according to the profile and if all required traceability links to related artifacts are present.

Imp#4: Library for safety case pattern: Extending the modeling tool with a library of safety case patterns should be investigated. This would support the re-use of best practices of the GSN models by the tool-chain.

These improvements serve as the basis for our research group's future investigations in the area of semi-formal safety modeling.

Conclusion and Outlook

The paper proposed a safety workflow that covers the concept phase of the automotive functional safety standard ISO 26262. The approach provides an overall view of the relevant semi-formal safety modeling artifacts. For the modeling of the safety artifacts, a SysML profile was introduced that supports the creation and management of the safety artifacts required in the safety workflow. An integrated safety analysis tool chain was demonstrated using the Enterprise Architect system modeling tool and APIS IQ FMEA for the Concept FMEA. The applicability of the approach was demonstrated using an automotive use case of a battery system of a HEV powertrain, which showed that the approach is generally suitable for enhancing the quality of the artifacts in the safety workflow and the safety argumentation.

The combination of a safety-oriented workflow and semi-formal modelling helps the relevant stakeholders perform safety engineering activities in a systematic way, as required by various standards such as ISO 26262.

Further investigations on the topics of system modeling with respect to functional safety activities should address the extension of the presented approach to the technical solution level (System/SW/HW description). The data exchange of safety analysis methods (e.g. System FTA/FMEA and FMEDA) should be extended. In addition, predefined safety case patterns should be made available by adding a library extension of the modeling tool.

References

1. Thomas, Nolte, Hansson Hans, and Bello Lucia Lo. “Automotive communications-past, current and future.” Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on. Vol. 1. IEEE, 2005.
2. John, Fitzgerald, Larsen Peter Gorm, and Verhoef Marcel. “From Embedded to Cyber-Physical Systems: Challenges and Future Directions.” Collaborative Design for Embedded Systems. Springer Berlin Heidelberg, 2014. 293-303.
3. ISO 26262 - “Road vehicles - Functional safety”, Part 1-10, November 2011.
4. ISO 12405 - “Electrically propelled road vehicles - Test specification for lithium-ion traction battery packs and systems”, 2011.
5. IEC 61508 - “Functional safety of electrical/electronic/programmable electronic safety-related systems.” International Electrotechnical Commission, 2 edition, 2010.
6. Object Management Group (OMG): OMG Systems Modeling Language (OMG SysML) Version 1.2. OMG Document Number formal/2010-06-01, 2010.
7. Yanja, Dajsuren, et al. “Automotive ADLs: a study on enforcing consistency through multiple architectural levels.” Proceedings of the 8th international ACM SIGSOFT conference on Quality of Software Architectures. ACM, 2012.
8. Eric, Andrianarison, and Piques Jean-Denis. “SysML for embedded automotive Systems: a practical approach.” Conf. on Embedded Real Time Software and Systems. 2010.
9. Piques, J. D., and Andrianarison. E. “SysML for embedded automotive systems: lessons learned.” Interfaces 3 (2012): 3b.
10. Martin, H., Baumgart, S., Leitner, A., and Watzenig, D., “Challenges for Reuse in a Safety-Critical Context: A State-of-Practice Study,” SAE Technical Paper 2014-01-0218, 2014, doi:10.4271/2014-01-0218.

11. Ericson, Clifton A. "Hazard analysis techniques for system safety." John Wiley & Sons, 2005.
12. Leveson, Nancy G., and Diaz-Herrera Jorge. "Safeware: system safety and computers." Vol. 680. Reading: Addison-Wesley, 1995.
13. VDA. Quality management in the Automotive Industry Volume 4 Chapter Product and Process FMEA, 2nd edition December 2006.
14. FORD, FMEA Handbook Version 4.1, Ford Motor Company 2004.
15. Wilson, S. P., Kelly Tim P., and McDermid John A.. "Safety case development: Current practice, future prospects." Safety and Reliability of Software Based Systems. Springer London, 1997. 135-156.
16. Kelly, T., "A Systematic Approach to Safety Case Management," SAE Technical Paper 2004-01-1779, 2004, doi:10.4271/2004-01-1779.
17. Kelly, Timothy Patrick, "Arguing safety: a systematic approach to managing safety cases", University of York, 1999.
18. Kelly, Tim P., and McDermid John A., "Safety case construction and reuse using patterns", Safe Comp 97. Springer London, 1997. 55-69.
19. Menon C, Hawkins R, McDermid J., "Interim standard of best practice on software in the context of DS 00-56 Issue 4. Technical Report SSEI-BP-000001", Software Systems Engineering Initiative, York, https://ssei.org.uk/documents/. Accessed 5 October 2009.
20. Hawkins R., and Kelly T.. "A Software Safety Argument Pattern Catalogue." Department of Computer Science The University of York
21. GSN Community "GSN COMMUNITY STANDARD - VERSION 1", http://www.goalstructuringnotation.info, 2011.
22. Yiannis, Papadopoulos, Parker David, and Gran C.. "Automating the failure modes and effects analysis of safety critical systems." *High Assurance Systems Engineering, 2004. Proceedings. Eighth IEEE International Symposium on*. IEEE, 2004.
23. Yiannis, Papadopoulos, Parker David, and Grante Christian. "A method and tool support for model-based semi-automated failure modes and effects analysis of engineering designs." Proceedings of the 9th Australian workshop on Safety critical systems and software-Volume 47. Australian Computer Society, Inc., 2004.

Acknowledgments

The research leading to these results has received funding from the ARTEMIS Joint Undertaking under grant agreements of nSafeCer n○295373. The authors would like to acknowledge the financial support from the Austrian Research Promotion Agency (FFG) of the Federal Province of Austria for the project in which the research results presented above were achieved. Furthermore, the authors would like to acknowledge APIS GmbH for their support by their software tool, APIS IQ FMEA PRO.

ISSN 0148-7191

http://papers.sae.org/2015-01-0273

Type Paper Number Here

From Natural Language to Semi-Formal Notation Requirements for Automotive Safety

Author, co-author (Do NOT enter this information. It will be pulled from participant tab in MyTechZone)

Affiliation (Do NOT enter this information. It will be pulled from participant tab in MyTechZone)

Abstract

The standard ISO 26262 stipulates a “top-down” approach based on the process “V” model, by conducting a hazard analysis and risk assessment to determine the safety goals, and subsequently derives the safety requirements down to the appropriate element level. The specification of safety goals is targeted towards identified hazardous events, whereas the classification of safety requirements does not always turn out non-ambiguous. While requirement formalization turns out to be advantageous, the translation from natural language to semi-formal requirements, especially in context of ISO 26262, poses a problem. In this publication, a new approach for the formalization of safety requirements is introduced, targeting the demands of safety standard ISO 26262. Its part 8, clause 6 (“Specification and management of safety requirements”) has no dedicated work product to accomplish this challenging task. The five levels of requirements for writing safety requirements are distributed throughout the standard, increasing the probability of misapplication. For these reasons, a dedicated requirement template is proposed. It is applicable for writing new or checking existing requirements, independent of any tool. By reviewing a number of industrial relevant use cases the applicability of the new template is verified and its effectiveness is demonstrated. Furthermore, a semi-formal notation technique is shown to express these formalized requirements, including their associated attributes and resulting relationships. By following the proposed approach, we meet the obligations of ISO 26262 to write e.g. unambiguous, consistent, verifiable, and complete requirements. In the end, this has the potential to dramatically reduce the probability of systematic failures during development of automotive embedded systems.

Introduction

With ever increasing reliance on electric and electronic components contributing to the functional safety of next generation vehicles, a single malfunction can initiate a costly product recall. Automotive E/E systems perform highly networked functions with large numbers of features in numerous product variants. Dense system interaction increases complexity beyond human understanding. Generally speaking, these properties are considered to be the main source for systematic faults. They do not only affect newly developed systems, like complex driver assistance systems. Also well-known systems often bear a non-negligible safety threat, especially when they are considered in a different context than initially intended.

Besides the fact that writing requirements is non-trivial, the number of requirements needed to specify a technical product with all its features, variants and safety is usually large. The automotive market demands more product functions than ever before. This complexity increases the number of malfunctions. Typical modern cars are available in a broad range of variants. Different propulsion systems and combinations thereof are available. For example, hybrid electric vehicles or pure electric vehicles require sophisticated electronic controls. Advanced driver assistant systems as well as active and passive safety systems provide additional comfort and protect passengers. The amount of functions provided by all these various systems is the reason where the term complexity stems from. Each of these single systems is generally well understood today, but the interactions with and cross-relationships to other systems may not be fully understood and this is critical for functional safety.

Vehicles are developed by a multi-level architectural hierarchy, where each level has different technology (and intellectual property) characteristics. At the top level, safety critical automotive product development is initiated with a comprehensive and well-coordinated concept phase. Its hazard analysis and risk assessment evaluates the risks the product’s users and operators face in various situations and where the product possibly contributes to this risk. One of the key outputs of the hazard analysis and risk assessment is a set of safety goals. Safety goals are top level safety requirements that should safeguard the system’s intended behavior in an implementation-independent way. Based on these safety goals, a tree of safety requirements emerges from top to bottom, including functional safety requirements, technical safety requirements, software safety requirements and hardware safety requirements. In general, requirements are often mistaken for specifications. Whereas the first reflect a black-box view on the system under development, the latter reflect a white-box view. All these aspects span a large space of formulation and interpretation for safety requirements across the multi-level architectural hierarchy. Natural language is by it's name the way we communicate and appears to be the most intuitive language available. Therefore it is considered as the primary means for writing requirements. At the same time it provides the whole spectrum of linguistic diversification, which motivates several different approaches for requirement formalization and allows systematic failures to be introduced.

The approach described within this work has been created within the VeTeSS[1] project. The VeTeSS consortium works on standardized tools and methods for verification of the robustness of automotive safety-relevant systems. To highlight the advantages of the proposed approach, an automotive airbag system is considered as a use case. This paper shows how to establish requirements ready for test and how these requirements are related to system design and architecture.

Problem Statement

Writing requirements for technical products is widely considered to be a time consuming activity to refine for use in development. The reasons for that are diverse. First of all, authors of requirements have a certain technological background that can unintentionally introduce a perspective reflected in the written requirements and derive into the specifications. Second, engineers who are focused on implementation of specifications may have a specific technology in mind, interpreting requirements in a domain specific way. Third, in the automotive supply chain, authors and readers (producers and consumers, respectively) of requirements are distributed across all stages of development, from OEM (original equipment manufacturer) level to tier *n* level (e.g. hardware suppliers). Finally, in this hierarchy, requirements are often the sole written work products of technical communication. Hence, requirements are expected to fulfill properties like unambiguousness, completeness, indivisibility, and comprehensibility, which are key issues in this field.

In functional safety, requirement engineering plays a major role due to the fact that all safety requirements address potential malfunctions of all these composite systems. This is considered as main motivation to produce sets of requirements with integrity. Several safety standards are aware of these challenges and state numerous criteria for writing requirements, e.g. IEC 61508 or ISO 26262. The majority of them do not state however, how these criteria should be met.

Objectives from ISO 26262

Properties of Safety Requirements

ISO 26262 represents the standard for functional safety of road vehicles, targeting passenger cars up to 3.5 tons of gross weight. It consists of 10 parts and covers the entire safety life cycle of electric and electronic systems, and their relevant associated systems. In its part 8, clause 6, the "Specification and management of safety requirements" is specified. In the present paper only safety requirements in context of ISO 26262 are in focus. Their unique feature is the direct or hierarchically linked assignment to a malfunction, which is relevant in terms of functional safety either directly or indirectly.

The ISO 26262 part 8, clause 6 defines the quality criteria for all safety requirements are the following: Atomic, unique, abstract, level defined (non-redundant, modular, structured, satisfied, and qualified). ISO 26262-part 3 defines the safety goals and functional safety requirements. Part 4, clause 6 defines the system level and likewise parts 5 & 6 provide the numerous characteristics for the specification of single safety requirements for their corresponding elements. Due to lack of definitions within the standard, some definitions are taken from [1].

First of all, unintentional ambiguity shall be avoided. Ambiguity is referred to as a commonly understood meaning of the requirement within a specific context. This is considered to be one of the primary sources for systematic failures, especially in the automotive supply chain when it comes to the exchange of requirements between different developers of OEMs and/or suppliers. Comprehensibility also contributes to this field, especially if engineers from different domains work together. Different domains typically use sets of well established vocabularies, which must be matched. Furthermore, requirements should be atomic. This means that one requirement should carry one single traceable piece of information. This drives the ability to verify it, without it another type of systematic failure can be introduced. Internal consistency ensures that one safety requirement contains no contradictions with another safety requirement, e.g. one requirement is for the safety goal and the second one a functional safety requirement. The property of feasibility demands that one single safety requirement can be mapped to a specific technical implementation. Finally, verifiability ensures that the standard's criteria have been met and that proper evidence can be generated proving the fulfilment of safety requirements. In order to verify safety requirements, ISO 26262 recommends semi-formal verification methods for all ASILs, especially for ASILs C and D (ISO 26262, part 8, Table 1).

ISO 26262 also quotes characteristics for sets of safety requirements and their management. They should be hierarchically structured, in order to reflect the standard's development phases (see ISO 26262-8, Figure 2). Furthermore, sets of safety requirements should be grouped together, according to the system's architecture. The property of completeness means that the safety requirements at one level fully implement all safety requirements of the previous level. Contrary to internal consistency, external consistency means that multiple safety requirements do not contradict each other. Redundancy between safety requirements shall be avoided in general, that is duplication of information within any level of the hierarchical structure. This is the purpose of the attributes to be documented on only one level of requirements. Finally, maintainability ensures that requirements can be modified or extended. This includes the introduction of new versions of safety requirements but also the addition or removal of safety requirements to or from the set. Additional to these properties, safety requirements shall also be traceable. That means that a logical link shall be established to each source of a safety requirement at the upper hierarchical level, the lower hierarchical level, or to its realization in the design. To enforce traceability of verification results, each safety

[1] http://www.vetess.eu

requirement shall also be linked to its corresponding specification of verification.

Besides these definitions found in part 8, clause 6, additional requirements are specified throughout the standard. An example thereof is part 3, clause 8.4.2.3, which states that functional safety requirements shall be specified by the function to manage deviations and include also operating modes, fault tolerant time intervals or safe states. Another example can be found in part 5, clause 7, where hardware safety requirements need to define five aspects, a) behavior of the element, b) safety mechanisms for the internal faults, c) safety mechanisms for the external faults from other elements, d) the safety mechanism for the interactions from the other elements and e) the safety mechanism to cover latent faults.

One can conclude that the standard does not reflect an integrated view on the construction of safety requirements. An extensive list of properties to obey when writing requirements can be found in [1], [2]. In this paper, focus is on ISO 26262, part 8, clause 6, entitled “Specification and management of safety requirements” for each of the element levels through the standard. This work focuses on methodology, how to specify safety requirements and to fulfill all relevant criteria. Numerous tools are already available on the market, targeting ISO 26262. In context of this paper, a tool independent solution should be elaborated.

The Rationale of Safety Requirements

Safety requirements differ from traditional product or process requirements as they shall be (a) targeted towards one or more concrete malfunctions and (b) assigned to at least one architectural element with the purpose to mitigate the output failure of the architectural element.

Malfunctions from (a) are addressed through the formulation of safety goals. In order to fulfill (b), ISO 26262 knows five subsequent levels where safety requirements must be formulated. The “hazard analysis and risk assessment” must be conducted in order to disclose potential safety risks to the item under development. The functional safety concept is intended to represent an -independent view on the product to manage the known failures. Its requirements are named “functional safety requirements”. They are derived from their corresponding safety goals to identify needed safety measures to avoid these imminent dangers so that they can be mitigated somewhere in the “Item’s architecture. “Technical safety requirements” are derived from their corresponding functional safety requirements. They are allocated to system elements for implementation by the system design. Finally, the hardware and software safety requirements are derived from either their corresponding or assumed technical safety requirements.

Requirements are defined down to the component level (hardware parts and/or software units, according to ISO 26262, part 1).The component is usually the element that is removed during a repair event, when the car does no longer operate as specified.

If this hierarchical approach is strictly followed, each safety requirement contributes to one or more safety goals, hence contributes indirectly to the mitigation of one or more hazards by the prevention of faults becoming errors, and errors subsequently causing failures. This highlights the importance of a properly conducted hazard analysis and risk assessment during the concept phase.

For the fulfillment of (b), ISO 26262 knows several architectural elements. The item implements a function at vehicle level and consists of a system or an array of systems. Safety goals apply to this level. Functional safety requirements apply to this level as well and address intended functions. A system relates at least a sensor, a controller and an actuator with one another. Technical safety requirements apply to the system level, they are allocated to system elements. Components are composed of one or more hardware parts and/or software units. Hardware safety requirements and software safety requirements apply to this level.

Writing Safety Requirements

The most intuitive way of writing requirements is using natural language. It is the best way to start to define the requirements in the elicit and gather process. During the clarification and analysis process of requirements management, a method is needed to convert this natural language into semi-formal to complete two things at once. On one hand improve the technical quality, and on the other hand reduce the duplications which are process anomalies that generate even more systematic failures at a later stage.

The template defined by the VeTeSS project is used as the initial verification report. For the technical contents it checks if a set of requirements meets the standard’s requirements from part 8, clause 6. First of all, a distinction between requirements, specifications and implementation is introduced. A requirement defines the expected behavior of an element. The safety goal is always a negative requirement or objective, while all the other safety requirements are those that can be verified. A specification is defined as a piece of information addressing a requirement. The specification contains key information how the requirement should be realized, aided by a certain technology or technical solution. An implementation is defined as a piece of technology realizing a specification. In terms of ISO 26262, this refers to all kinds of code (software, hardware description languages), hardware, as well as elements of other technologies (e.g. mechanical domain).

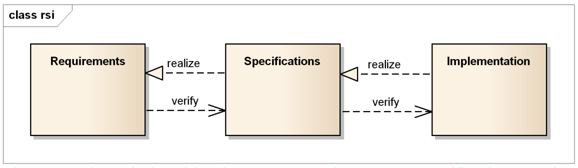


Figure 1: The relationships between requirements, specifications and implementation.

Contributions of this Work

All in all, the main challenge in writing safety requirements is to consider their underlying hierarchical levels and their target architectural elements. Both of these aspects emerge during the concept phase. First, the results of a hazard analysis and risk assessment help to determine a hierarchical system design. Second, preliminary architectural assumptions are taken, in order to establish a first draft of the item under development. Both approaches together should lead to a set of usable safety requirements.

Requirements are often mistaken for specifications. However, there should be a clear distinction between these both, in order to maintain the environment of a problem space (the formulation of requirements and safety requirements, stating *what* shall be achieved) and a solution space (the formulation of specifications, stating how the actual implementation shall fulfill the requirements). This concept is well known for the development of embedded systems, and refers to the black box/white box approach introduced earlier. While requirements typically describe the black box behavior, specifications describe the white box behavior.

In this work, we propose 3 contributions to the state-of-the-art in writing safety requirements for automotive applications, their formalization, and transformation into semi-formal notation.

1. We propose an ISO 26262 compatible template for writing safety requirements. This template contains a number of patterns which allow the establishment of cross-relationships to artifacts of other problem domains in the same context (architecture, behavior, etc.)

2. We propose a SysML profile containing stereotypes and relationships for modeling safety requirements according to the template.

3. We present a method for analysis of these gathered safety requirements, which helps during review and assessment steps.

A use case from the automotive domain highlights the benefits of the approach described within this work.

Related Work

The relationships between requirements development, verification, and validation are elaborated in [3]. The authors highlight the differences between those terms, contributing to the common understanding and increasing the probability of success of future system designs.

The authors of [1] provide a good overview to different requirement writing and formalization techniques. The method of boilerplates for a more formalized way of writing requirements is also introduced for the first time.

In [4] requirements engineering is identified as one of the crucial issues of automotive software development processes.

Systematic failures occur when the supply chain does not provide consistent and traceable product requirements from “top down” [5].

Stirgwolt [6] describes the shift from a quality oriented management to a safety work culture. Writing requirements is not the first step of product development, which is described as a key barrier for execution of this shift.

The thesis of [7] is dealing with the aspect of requirements ambiguity in context of ISO 26262. This work has a strong focus on the exact formulation of safety requirements utilizing a domain ontology and constrained natural language in terms of boilerplates.

In [8] the domain ontology design tool is elaborated. It semi-automatically transforms natural language requirements into semi-formal boilerplate requirements.

In [9] a semantic guidance system is introduced which assists requirements engineers with capturing requirements using a semi-formal representation.

Requirement templates represent guidelines for writing requirements [10]. They can be seen as a blueprint for the description of functionality and capabilities. The template described consists of several sections, each describing a special type of requirement and its variations.In [11] and [12] a unified requirements modelling language is proposed. It supports system and process modeling, danger modeling, feature modeling and goal modeling aspects. It is implemented as a UML profile on top of a well-defined meta model. Several implementations exist which are based on standard UML tools.

A systematic method for the definition of domain specific languages using UML profiles is given in [13]. This includes the definition of a domain meta model and a UML profile.

Solution

Why use patterns?

To save development time and reduce development costs, while achieving a reasonable quality for functional safety are motivations to use patterns. Let us address the “quality” aspect of requirements. ISO 26262 part 8, clause 6 defines the quality characteristics, for each requirement. The problem each project must resolve is how effective is it able to convert the natural language requirements into semi-formal requirements. Once this is done it is easier to convert into a model based development saves time and development costs while avoiding systematic failures to being introduced into the implemented design of an element. Currently the ISO 26262 has not released any *template*, *patterns* or *boilerplates*, yet the raw material is available. Many Artemis projects are evaluating and defining new methods, artifacts or tools to help support the improvement of state of the art. The ISO 26262 has industry specific requirements which need to be developed to achieve the needs of the market. This paper demonstrates how easy it can be to check the quality with proper and effective template, clear patterns and concise boilerplates. In order to address the targets highlighted previously, an organizational structure for requirements must be defined. It should be able to purport the frame conditions for our approach. For this reason we introduce the terms template, pattern, and boilerplate. A template is defined as a product specific or standard oriented guideline for writing requirements. Furthermore, it serves as a container for requirement patterns. For example, it may be implemented as a spreadsheet in response to a collection of requirements. A pattern defines the attributes (e.g. interfaces) and loose syntax describing what shall be used to construct linguistic sentences in terms of semi-formal notation. One pattern specifies the numerous attributes which a sentence must include, in order to comply with the pattern. A pattern however may not specify a strict linguistic syntax for exact formulation of sentences, covering all aspects and possibilities of natural language. It aids the distributed development of the Item to derive the requirements to the allocated elements. A single boilerplate is defined as a pre-defined requirement sentence containing placeholders. One boilerplate defines the strict syntax of a single requirement. During the process of writing requirements, its placeholders are completed with corresponding information. The textual form of the requirement can be generated when needed, by merging the information with the boilerplate. This is shown in [1], where several advantages are enumerated. For example, by separating the information from the boilerplate, an independent modification of requirement expressions is possible. Another example thereof is the collection of identical placeholders, which allows for easy sorting and filtering on specific information.

The relationships between template, pattern and boilerplate are shown in Figure 2. The paper at hand defines its scope on templates and patterns. Boilerplates are not in focus of this work. There are several reasons for that. First of all, it is intended to fill the gap for a work product according to ISO 26262, without purporting exactly how requirements should be written. Second, in the automotive industry, palettes of boilerplates are collected and classified as different ways of expressing certain kinds of requirements. These palettes of boilerplates may be defined differently from OEM level down to supplier levels. In many cases it does not necessarily seem reasonable to introduce entirely new boilerplates targeted at functional safety. Much more, existing boilerplates may be tailored to comply with the patterns related to functional safety from this paper. Third, other projects have dealt with or are dealing with the formulation and analysis of boilerplates. Examples thereof are the CESAR project and the succeeding CRYSTAL project.

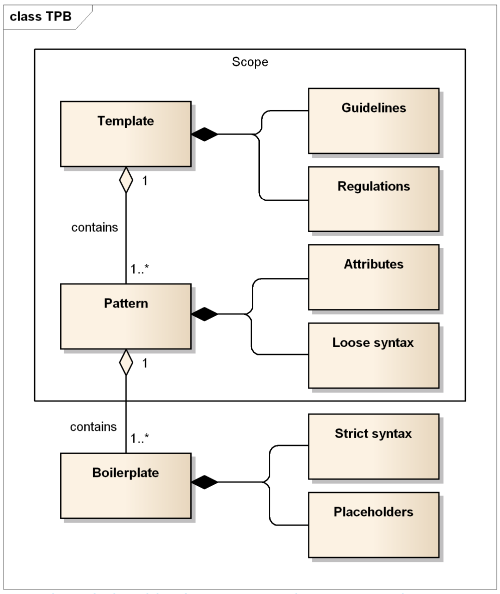


Figure 2: The relationships between requirement templates, patterns, and boilerplates, as they are defined in the VeTeSS project

Structure of the Template

The proposed template is organized as follows. First of all, it demands some meta-information, like the name of the requirement author or assessor, the baseline requirement document or work product it assesses, the item, element or product to be sold and project- or use case-relevant information. A (simplified) copy of the spreadsheet is given in the Appendix.

In vertical direction, it reflects the hierarchy of safety goals and safety requirements according to ISO 26262. At the same time, this hierarchy can be related to the architectural structures of the standard. Safety goals define the objectives of the item to avoid malfunctions which could lead to harm which are defined during the concept phase. Functional safety requirements convert these negative requirements into positive or feasible requirements to be performed by the item at vehicle level. Technical safety requirements are derived and address the systems according to ISO 26262. Software safety requirements define the behavior needed for the software components .Hardware safety requirements define the behavior for the hardware components. For ISO 26262 compliant development, it is worth to note that this requirement hierarchy must always be constructed otherwise a systematic failure could be introduced. Naturally, this happens in a top-down approach. But also during the process of re-constructing missing requirements on a higher level, a bottom-up approach requires at each level the assumed requirements to be re-addressed or re-evaluated in the correct hierarchy or sequence. Furthermore, each distributed developer must process their element on the proper hierarchy level. Therefore each level may contain information about organizational roles and responsibilities according to ISO 26262 part 2, entitled “Management of functional safety”.

In horizontal direction, the template proposes a number of attributes. Subsets of these attributes are subsequently used for the construction of requirement patterns. The attributes are grouped in two sections.

The left section is intended to specify external safety behavior, referred to as black box behavior or demand. It targets the respective architectural element of the affected hierarchical level. The right section is intended to specify internal safety behavior, referred to as white box behavior, or reaction. It targets the contents of the respective architectural element of the affected hierarchical level. In the following, the attributes of external safety behavior, which are used for the construction of requirement patterns, are discussed.

Requirement type: The requirement type specifies the type of the safety requirement. For this, the five options are available: Safety goal or top-level safety requirement, functional safety requirement, technical safety requirement, as well as hardware and software safety requirement.

Action: The action attribute specifies the verb of the safety requirement.

Malfunction: The malfunction attribute refers to a behavior output failure of the hierarchy.

Function: The function attribute represents the execution logic.

Demand actor: The demand actor attribute stands for the requestor of a function.

Reactor element: The reactor element attribute stands for the receiver of the output of a function.

Operational condition: The operational condition attribute is defined as the state of an architectural element. This state depends on the hierarchical level and must be defined accordingly.

Automotive safety integrity level (ASIL): This attribute is used for process tailoring, as a more stringent level requires additional efforts in order to safeguard the associated behavior.

In the following, the attributes of internal safety behavior, which are used for the construction of requirement patterns, are discussed.

Propagating known failures: This attribute tries to cover output deviations of the affected item or element. It determines differently than defined behavior, opposite to specified intended behavior.

Safe state 1: This attribute refers to safety measures realized through redundancy, with the intent to detect and control faults occurring in the affected element.

Time: The attribute of time is defined differently on each hierarchical level. The declaration of a value however is indispensable for specification of internal behavior.

Architectural elements: The definition of architectural elements from an internal (white box) point of view. This also includes information how the contained architectural elements are related to each other.

Safe state 2 & 3: The external safety measures define the safe states 2 and 3. Safe state 2 is referred to as warning level with fault detection. Safe state 3 is referred to as emergency level including degradation of the elements intended function.

Interfaces: The interfaces attribute defines the interfaces between the architectural elements as well as elements of other technologies.

Laws & Regulations: Finally, the laws and regulations attribute allows the specification of applicable legal requirements and standards.

These attributes primarily target the concept phase, and product development at the system, hardware and software levels. Depending on the scope of the developer on a specific level, the number of attributes may be increased to include other level specific aspects. In order to enforce a successful exchange of requirements at the interfaces between OEMs and the different suppliers, we suggest the previously introduced attributes as a minimum requirement.

Structure of Patterns

In this section the organization of patterns for safety requirements is elaborated.

Safety Goal requirement pattern: A safety goal represents the top-level safety requirement. It must be formulated negatively. Hence, the action attribute automatically turns to *avoid* for all safety goals. The malfunction attribute is used to express the hazardous behavior of the item. Furthermore, it uses the reactor element, operational condition, and ASIL attributes.

Functional Safety Requirement pattern: A functional safety requirement is derived from at least one safety goal. It is formulated positively. The action attribute is used to represent the necessary verb. The function attribute is used to specify the intended functionality for the vehicle level. The reactor element and demand actor attributes are used to specify the sensing and actuating systems. Furthermore, the operational condition and ASIL attributes are specified as well.

Technical Safety Requirement pattern: A technical safety requirement is derived from at least one functional safety requirement. It is formulated positively. The action attribute is used to represent the necessary verb. The function attribute is used to specify the intended functionality for the system level.

Software Safety Requirement pattern: A software safety requirement is derived from at least one technical safety requirement. It is formulated positively. The action attribute is used to represent the necessary verb. The function attribute addresses a subfunction at software level. The reactor element attribute targets the receiver of software generated outputs. The demand actor attribute and the ASIL attribute are used. On the software level, must be systematically defined to support all operational conditions as specified due to the fact that software as a provider of data flow control utilizes the underlying hardware and operates on the system level.

Hardware Safety Requirement pattern: A hardware safety requirement is derived from at least one technical safety requirement. It is formulated positively. The action attribute specified the necessary verb. The function attribute specifies the subfunction at hardware level. The reactor element attribute is used to characterize the output of the subfunction. At hardware level, no demand actor is specified, due to the fact that hardware is seen as a provider for data management functions.

Translation from Natural Language to Semi-Formal Notation

Typically, requirements are captured, elicited, and collected using requirements management software. Good examples thereof are tools like Medini Analyze[2], DOORS[3], or PTC

[2] http://www.ikv.de/medinianalyze

[3] http://www.ibm.com/software/products/de/ratidoor

Integrity[4]. The introduced template and its patterns are generic, that means tool independent. They may be used twofold:

First application is the evaluation of existing safety requirements. The second application is the generation of new safety requirements. Experience has shown that with an increasing number of safety requirements, advanced methods are necessary in order to keep up with the increasing complexity. The complexity here emerges from four different operations, which recur at all hierarchical levels during the automotive development process. They are defined as follows.

- Collect: Safety requirements shall be collected from all relevant sources and at all relevant hierarchical levels.
- Analyze: The requirements analysis represents a quality check for technical contents. Basically this refers to the completeness and soundness of all used attributes. (Note it determines anomalies such as gaps, overlaps and incomplete allocations for traceability due to the logic for a safety goal)
- Repair: If any anomalies are identified during the analysis process, they of course must be corrected. It is worth to note that a detected anomaly must be corrected at the right hierarchical level. Anomalies in requirements are classified as systematic faults; therefore they must be fixed at their appropriate element level.
- Execute: The collected, analyzed, and corrected sets of requirements are released for a baseline. Subsets of these requirements may be communicated to the next lower hierarchical level for further development activities.

For these tasks, plain text, lists, or spreadsheets are insufficient. Those solutions are considered two-dimensional, and suffer from low traceability and a lack of modularity. Besides that, these approaches are susceptible to systematic faults, which may be introduced during changes and modifications.

To overcome these issues, formalization has shown to be beneficial. This is recognized by ISO 26262, part 6 for software architectural design, software unit design, and software verification tasks. The extension of these ideas to other parts of ISO 26262, like concept phase or system level is subject to current projects in industry and academia. In this paper, the conversion from natural language to semi-formal notation is covered, while the option persists to advance to full formal notation later on.

Semi-formal notation in general refers to a representation with well-defined language syntax and an informal defined language semantics. This stands in contrast to full formal notation methods, which are more stringent and have a mathematically defined language syntax and semantics. The term informal refers to a less stringent way of notation, where e.g. natural language with arbitrary language symbols is used. With the definition of patterns, featuring fixed intentional attributes as building blocks for the representation of desired semantics in the context of functional safety, the notation of safety requirements already stepped up from an informal notation to a structured language notation, which is considered semi-formal.

. The *Systems Modelling Language* (SysML) is standardized in an OMG standard [14], and knows a total of nine different diagrams, depicting several aspects of systems engineering. One of these diagrams is entitled *Requirement Diagram*. It allows the collection, organization, and structuring of arbitrary kinds of requirements. The latest SysML standard version 1.3 knows only one type of requirement. It does not distinguish between e.g. functional and non-functional requirements, or safety and non-safety requirements. As SysML is based on UML 2, it also inherits its profiling mechanisms. A profile provides a generic extension mechanism for customization of models. A profile consists of stereotypes, tag definitions and constraints, which are applied to e.g. classes or attributes.

To demonstrate the benefits of the proposed requirements template and to highlight the advantages of semi-formal notation, the template and its patterns are to be transformed into a SysML profile, extending the capabilities of SysML according to ISO 26262 and enabling a reuse of already defined safety artefacts. Such artefacts may be preliminary architectural assumptions or architectural designs in shape of a Block Definition Diagram (bdd) or Internal Block Diagram (ibd). Functions and malfunctions, which were identified in course of the hazard analysis and risk assessment during concept phase, may be available in shape of a Use Case Diagram (uc).

As each of the patterns represents a safety requirement at a specific hierarchical level of ISO 26262, it seems reasonable to extend the *SysML1.3::Requirement* stereotype with the five types of safety requirements known by ISO 26262. Each of these earns a tagged value indicating its minimum ASIL level. Furthermore, a *derive* relationship stereotype is defined, aiming at the right derivation depending on the hierarchical level of the safety requirement. Part of the resulting profile is shown in Figure 3.

[4] http://ptc.com/product/integrity

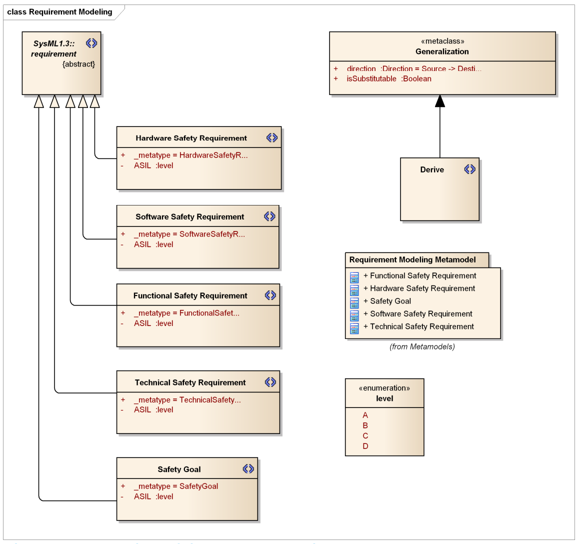

class Requirement Modeling
SysML1.3:: requirement {abstract}
«metaclass» Generalization
+ direction :Direction = Source -> Desti...
+ isSubstitutable :Boolean
Hardware Safety Requirement
+ _metatype = HardwareSafetyR...
- ASIL :level
Derive
Software Safety Requirement
+ _metatype = SoftwareSafetyR...
- ASIL :level
Requirement Modeling Metamodel
+ Functional Safety Requirement
+ Hardware Safety Requirement
+ Safety Goal
+ Software Safety Requirement
+ Technical Safety Requirement
(from Metamodels)
Functional Safety Requirement
+ _metatype = FunctionalSafet...
- ASIL :level
Technical Safety Requirement
+ _metatype = TechnicalSafety...
- ASIL :level
«enumeration» level
A
B
C
D
Safety Goal
+ _metatype = SafetyGoal
- ASIL :level

Figure 3: Extension of the SysML requirements concept.

Additionally, it makes sense to define the architectural elements according to the standard. These extend the block concept of SysML and include the *Item* as well as *System*, *Sensor*, *Controller*, *Actuator*, *Hardware part* and *Software Unit*. To enable multi-dimensional requirement adoption, the template's attributes must be present within the profile as well. Extending architectural element classes would dramatically increase the number of required relationships on diagrams and system models. For this reason, for each of the template's attributes a stereotype is created which extends the *Dependency* relationship. The result of these steps is a SysML profile reflecting the hierarchical levels of ISO 26262 and all attributes defined within the requirement template's patterns. It may be implemented in any UML tool capable of SysML modeling and UML/SysML profile definition.

Automotive Airbag System Use Case

A use case from the automotive domain highlights the application of the proposed requirements template and the semi-formal requirements modelling approach. We consider an airbag system which can nowadays be found in every passenger car. It is a *system* in terms of ISO 26262. The corresponding *item* is the passenger restraint system. It involves components of other technological domains of the vehicle as well, like seats or safety belts for example. However, these may also be controlled through electric and electronic systems. The air bag system relates sensors, actuators and a controller with each other. The controller consists of hardware and software elements. These are an air bag system on chip (SoC) hardware element, and a main micro-controller (uC) with corresponding software. The airbag system consists of various sensors. The most obvious ones observe different crash scenarios on vehicle level, e.g. upfront or side-impact crash scenarios. The sensors may use different technologies for that. The uses of G-force sensors or pressure sensors are two possibilities. Furthermore, additional sensors may be used to detect the present number and location of passengers within the vehicle. The airbag system also consists of numerous actuators. The actuators used to deploy automotive airbags are usually called squibs or gas generators. They are used to inflate air bags in vehicles intended for active passenger protection during collision scenarios.

Functional Description

An Airbag system has sensor components to capture the crash event. This is achieved by constant monitoring of sensor values. The controller needs to collect the operational condition and confirms the crash event so that an airbag fire signal can be generated. Inside the controller, the SoC performs an analog to digital conversion of the sensor data. It communicates with the uC using an serial peripheral interface (SPI). The uC decides if and when the airbag actuators shall be deployed and requests deployment from the SoC, which in turn drives a high analogue current to the gas generator resistors. The emerging heat of these resistors activates a chemical reaction, resulting in airbag inflation, protecting the passenger within the vehicle. Experience has shown that airbag systems typically perform of up to 20 networked functions, depending on the features and configuration of the vehicle. For this work focus is on two typical use cases which can be related to the two following safety goals

1) to avoid the lack of deployment of the airbag during a crash event, classified according to the hazard and risk analysis with ASIL A, and

2) to avoid the deployment of the airbag during normal driving conditions (no crash event), classified according to the hazard and risk analysis with ASIL D.

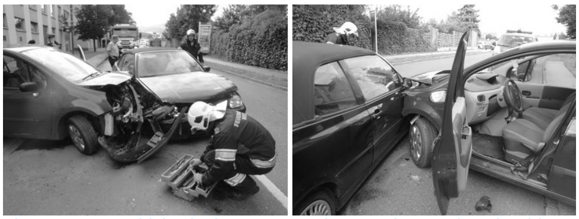


Figure 4: Two vehicles in the same crash scenario. One has the driver airbag inflated, in the other the driver airbag did not deploy (right). Image courtesy fire brigade.

A real world crash scenario is shown for illustrative purposes in Figure 4. The two photographs were taken from different viewing angles. Both vehicles are involved in the same crash scenario, but only one airbag did inflate to protect the driver. This raises questions if the requirements and subsequent specifications of these systems were correct.

Hardware Part

The controller or application specific integrated circuit must be defined by the expected “black box” behavior. This is usually defined in the hardware design specification, the communication protocol, the interfaces, the time, the voltage necessary to “fire” an airbag actuator or ignitor. The ability to read the sensor data, communicate with the micro controller to provide a fire signal and the controller then determines the conditions when to send the “fire” signal. Each depends on the item definition or application, the effectiveness of the system to detect and control all “known faults”.

Software Unit

The software components living within the microcontroller contain the execution logic for the sensor data conversion and evaluation. Evaluation information includes the logic for airbag deployment according to a broad range of factors. These include the impact signals transmitted by the sensors, the vehicle’s operational condition (e.g. cruise mode, standstill, etc.), the number and characteristics of occupants within the vehicle (e.g. presence, weight), as well as information about other systems (e.g. safety belt buckles), or possible child safety seats. The evaluation result is compared with the evaluation result of the hardware SoC. If the airbag system is in an actual crash scenario, it looks up the best passenger protection strategy and engages its single airbags accordingly.

Example: A derived Safety Requirement

In this section, an example is exercised from the top-level safety goal down to hardware and software levels. Note that this represents a minimal cut-out of a larger set of safety requirements. For a safety goal, we want to avoid the deployment of the airbag during no crash event (as defined in the previous section). According to the corresponding pattern, this gives:

Table 1: Safety goal pattern example.

Requirement Type	Action	Malfunction	Reactor element	Operational condition	ASIL
SG	avoid	Inadvertent deployment	airbag	all	D

From this negatively written safety goal we derive a functional safety requirement. It basically states that the airbag shall deploy in a crash scenario. This gives:

Table 2: Functional safety requirement example.

Requirement Type	Action	Function	Reactor element	Operational condition	Demand actor	ASIL
FSR	deploy	accident specification	airbag	normal vehicle	airbag ECU	D

From this functional safety requirement, a technical safety requirement is derived, introducing a technical solution to the problem, involving sensors, a controller and actuators. This gives:

Table 3: Technical safety requirement pattern example.

Requirement Type	Action	Function	Reactor element	Operational condition	Demand actor	ASIL
TSR	actuate	actuation for small statured adult	driver airbag	frontal crash scenario	dashboard control panel	D

Based on this technical safety requirement, hardware and software safety requirements must be derived. An example for a software safety requirement is given in the next table:

Table 4: Software safety requirement pattern example.

Requirement Type	Action	Function	Reactor element	Demand actor	ASIL
SSR	confirm	trigger fire signal	security peripheral	airbag ECU	D

Based on the previous technical safety requirement, an example for a hardware safety requirement is given in the next table:

Table 5: Hardware safety requirement pattern example.

Requirement Type	Action	Function	Reactor element	Operational condition	ASIL
HSR	send	redundancy	fire signal	accident scenario	D

To demonstrate the semi-formal modelling, the technical safety requirement from Table 3 is picked out. A SysML requirements diagram is chosen to depict the corresponding information, as seen in Figure 5. The technical safety requirement artefact is shown in the center. On the left and right hand sides, the requirements attributes are located, connected with the requirements artefact using the intended relationships. On the bottom, two subsequently derived hardware and software safety requirements are shown. The coherent requirement text in natural language is part of the technical safety requirement artefact and is generated manually.

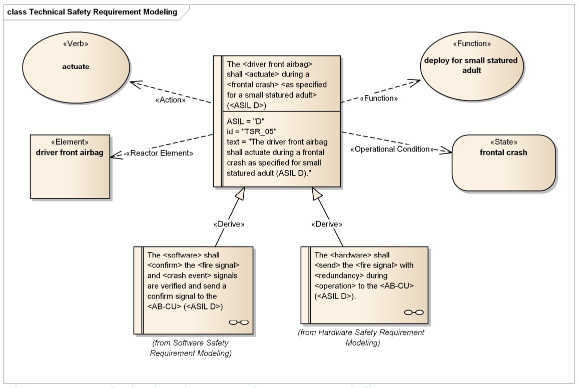

class Technical Safety Requirement Modeling
«Verb»
actuate
«Action»
The <driver front airbag> shall <actuate> during a <frontal crash> <as specified for a small statured adult> (<ASIL D>)
ASIL = "D"
id = "TSR_05"
text = "The driver front airbag shall actuate during a frontal crash as specified for small statured adult (ASIL D)."
«Function»
deploy for small statured adult
«Element»
driver front airbag
«Reactor Element»
«Operational Condition»
«State»
frontal crash
«Derive»
«Derive»
The <software> shall <confirm> the <fire signal> and <crash event> signals are verified and send a confirm signal to the <AB-CU> (<ASIL D>)
(from Software Safety Requirement Modeling)
The <hardware> shall <send> the <fire signal> with <redundancy> during <operation> to the <AB-CU> (<ASIL D>).
(from Hardware Safety Requirement Modeling)

Figure 5: Technical safety requirement modeling.

Evaluation and Discussion of Benefits

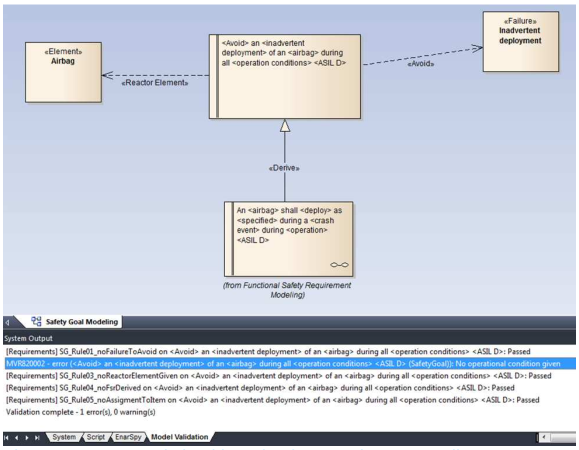
The VeTeSS project selected products from the different element levels to represent the typical state of the products to be assessed. The project focuses on how to generate the correct verification methods and tools to support ISO 26262 and drive further the state of the art. System, software and hardware elements were included in the sample. The partners used their natural language safety requirements. They applied all the ISO 26262 requirements in all different perspectives. One perspective this paper wishes to highlight is how well were the natural language requirements written by the expert: Without a ruler to measure the quality then as engineers it is nearly impossible to guess the quality.

So this team took the ISO 26262 requirements, drafted a method and semiformal template to check these natural language requirements. From the sample of VeTeSS requirements (note that the actual requirements are project confidential) over 30% of the sample assessed missed at least one or more attributes of the level of the safety requirements as defined by the ISO 26262. Typical errors were incomplete or inconsistently written requirements. Incompleteness was due to the lack of the level, or attributes needed. Inconsistence could be found with if the author did not have the top down (or outside in) perspective. As innovation is an inside out development, then it can be additionally difficult to meet these "V" model requirements. The benefit of the "V"-model is to encourage a project to take a harder review to raise awareness of the first time right development. The requirement characteristics defined are the first level of assessment after the intuitive approach of writing in the natural language.

Criteria for Evaluation

The use of the requirements template with its patterns has several advantages. First of all, ambiguity for single safety requirements can be reduced. Since each safety requirement at a level has a fixed set of relationships covering one attribute per artefact, redundant or contradictory pieces of information can be identified more easily. Indivisibility (atomicity) can also be addressed with this mechanism. Second, the allocation to at least one traceable architectural element increases the property of feasibility. From an inverse view, in the end one architectural element has a set of safety requirements assigned, which are subject to specification for implementation.

Third, the operational condition relationship or state artefact, respectively, addresses verifiability of the safety requirement. A corresponding test case affects the targeted architectural element. For entire sets of safety requirements, the template ensures traceability, as it enforces a hierarchical approach, where a safety requirement maintains a relationship to its parent safety requirement (which, in the end, is a safety goal as top level safety requirement). The utilization of SysML modeling techniques allows grouping of requirements in packages and visualization of sets on diagrams.


«Element»
Airbag
«Reactor Element»
<Avoid> an <inadvertent deployment> of an <airbag> during all <operation conditions> <ASIL D>
«Avoid»
«Failure»
Inadvertent deployment
«Derive»
An <airbag> shall <deploy> as <specified> during a <crash event> during <operation> <ASIL D>
(from Functional Safety Requirement Modeling)
Safety Goal Modeling
System Output
System
Script
EnarSpy
Model Validation

```
[Requirements] SG_Rule01_noFailureToAvoid on <Avoid> an <inadvertent deployment> of an <airbag> during all <operation conditions> <ASIL D>: Passed
MVR820002 - error (<Avoid> an <inadvertent deployment> of an <airbag> during all <operation conditions> <ASIL D> (SafetyGoal)): No operational condition given
[Requirements] SG_Rule03_noReactorElementGiven on <Avoid> an <inadvertent deployment> of an <airbag> during all <operation conditions> <ASIL D>: Passed
[Requirements] SG_Rule04_noFsrDerived on <Avoid> an <inadvertent deployment> of an <airbag> during all <operation conditions> <ASIL D>: Passed
[Requirements] SG_Rule05_noAssigmentToItem on <Avoid> an <inadvertent deployment> of an <airbag> during all <operation conditions> <ASIL D>: Passed
Validation complete - 1 error(s), 0 warning(s)
```

Figure 6: Automated checking of safety requirement attributes.

In a model based environment, the checking of such properties for consistency and completeness can be automated. This requires the formulation of rules, which are derived from the safety requirements meta model. To check for the most common properties on the white box view of safety requirements, approximately 30 rules are necessary. These are implemented in the used industry tool (Sparx Systems Enterprise Architect) within a safety requirement add-in. A screenshot of the process of automated rule checking can be seen in Figure 6. After the application of rules to the model, the engineer is confronted with a number of passed or not passed evaluation results, pointing out possible deficiencies of the requirements model.

Common mistakes

In the VeTeSS project, 6 use cases from the automotive industry from all levels of the supply chain are assessed. Each use case has an item definition according to the concept phase of ISO 26262, featuring the results of a hazard analysis and risk assessment, as well as preliminary system architecture. The 6 use cases include the previously introduced airbag system, but to achieve more a significant evaluation result, all use cases were taken into account for analysis. Based on these, safety goals and safety requirements are available. These are subject for evaluation according to the introduced requirements template. Hence, the template was applied to check on existing requirements. The most common mistakes are highlighted here.

Missing malfunctions: On the item level, a common mistake is to exclude the malfunction from the safety goal. This often coincides with the fact that safety goals are not written negatively and hence do not make use of the verb *avoid*. While the hazard is addressed, the malfunction of the E/E system triggering the hazard is not. This has large potential for systematic faults, as the safety goals are not complete then. Subsequently derived safety requirements possibly do not address the identified hazard anymore. During the subsequent derivation of safety requirements, the results of the hazard analysis and risk assessment must be available and constantly looked up. Repeated comprehension and interpretation introduces systematic faults. Such a procedure also conflicts with the idea of (sets of) safety requirements as a standalone work product.

Wrong hierarchical level: During interaction and exchange of requirements between suppliers, safety issues might be resolved at the wrong hierarchical level. This leads to safety requirements targeting the wrong architectural level, e.g. a safety goal targeting the item level, stating:

“Avoid critical cell voltage of the battery.”

does not address the item level anymore, but rather the hardware level as it refers to some kind of voltage level.

Missing operational condition: This affects all hierarchical levels. However, the causes of such mistakes are often found in the hazard analysis and risk assessment. For the safety goal:

“Avoid unintended vehicle acceleration.”

the assessment of unintended acceleration might result in e.g. different ASIL ratings for different operational conditions.

Missing reactor element: The template’s requirement patterns inherently support the construction of causal loops, as demand actors, functions, and reactor elements are basically the building blocks of such loops. Thus, missing a reactor element as in:

“Avoid inadvertent deployment during all operational conditions.”

may break such loops. It is not immediately clear which architectural element is the receiver of the specified function’s output, thus such mistakes increase the potential for systematic faults.

Conclusions

Two key points can be observed in course of this work.

One is that the ISO 26262 standard would be much further advanced with a clear and concise work product to define a “state of the art” safety requirement. A product suffers greatly without having a specific template to check the technical and process compliance of the conversion of the natural language requirements into semi-formal notation to avoid systematic failures to aid model based development. Otherwise several developers with different perspectives automatically introduce these systematic failures.

Second is that without a known method how to verify the entire range of faults; systematic, fault injection and random faults with a 100% branch coverage are not very effective or beneficial. The complexity of today’s automobile functionality supports the ISO 26262 standard’s requirement to shift to model based engineering. The selected tooling must be both consistent and complete to cover those faults which could cause harm at the item level. The automobile industry is a cost competitive business environment. Therefore the distributed developers are driven to find effective ways to integrate these new development techniques into their quality and business management systems.

Future Work

An open topic for further research activities is the even stronger formalization of safety requirements into e.g. boilerplates. It is expected that only a subset of large boilerplate collections is necessary to reflect the requirements patterns introduced in this work. Together with such a selection of boilerplates, an accurate metric for exact analyses would be in favor of safety engineers to improve the quality of safety requirements.

References

[1] E. Hull, K. Jackson, and J. Dick, *Requirements Engineering*, 3rd ed. Springer, 2011.

[2] K. Pohl and C. Rupp, *Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam - Foundation Level - IREB compliant*. US: O’Reilly, 2011.

[3] A. T. Bahill and S. J. Henderson, “Requirements development, verification, and validation exhibited in famous failures,” *Systems Engineering*, vol. 8, no. 1, pp. 1–14, 2005.

[4] M. Broy, “Challenges in automotive software engineering,” *Proceeding of the 28th international conference on Software engineering ICSE 06*, vol. 2006, p. 33, 2006.

[5] P. Stirgwolt, “Getting Automotive Safety Integration (ASIL) Level Right From The Top Down For The Standard ISO 26262,” *Inside Functional Safety*, 2010.

[6] P. Stirgwolt, “Effective management of functional safety for ISO 26262 standard,” *Reliability and Maintainability Symposium (RAMS)*, 2013.

[7] P. Sternudd, “Unambiguous requirements in Functional Safety and ISO 26262: dream or reality?,” 2011.

[8] S. Farfeleder, T. Moser, A. Krall, T. Stalhane, H. Zojer, and C. Panis, “DODT: Increasing requirements formalism using domain ontologies for improved embedded systems development,” *14th IEEE International Symposium on Design and Diagnostics of*

Electronic Circuits and Systems, pp. 271–274, Apr. 2011.

[9] S. Farfeleder, T. Moser, A. Krall, T. Ståalhane, I. Omoronyia, and H. Zojer, “Ontology-driven guidance for requirements elicitation,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2011, vol. 6643 LNCS, pp. 212–226.

[10] S. Robertson and J. Robertson, *Mastering the Requirements Process: Getting Requirements Right*, Third Edit. Addison-Wesley, 2012.

[11] J. Helming, M. Koegel, F. Schneider, M. Haeger, C. Kaminski, B. Bruegge, and B. Berenbach, “Towards a unified Requirements Modeling Language,” *Requirements Engineering Visualization (REV), 2010 Fifth International Workshop on*, 2010.

[12] F. Schneider, H. Naughton, and B. Berenbach, “A modeling language to support early lifecycle requirements modeling for systems engineering,” *Procedia Computer Science*, vol. 8, pp. 201–206, Jan. 2012.

[13] B. Selic, “A Systematic Approach to Domain-Specific Language Design Using UML,” *10th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC’07)*, pp. 2–9, May 2007.

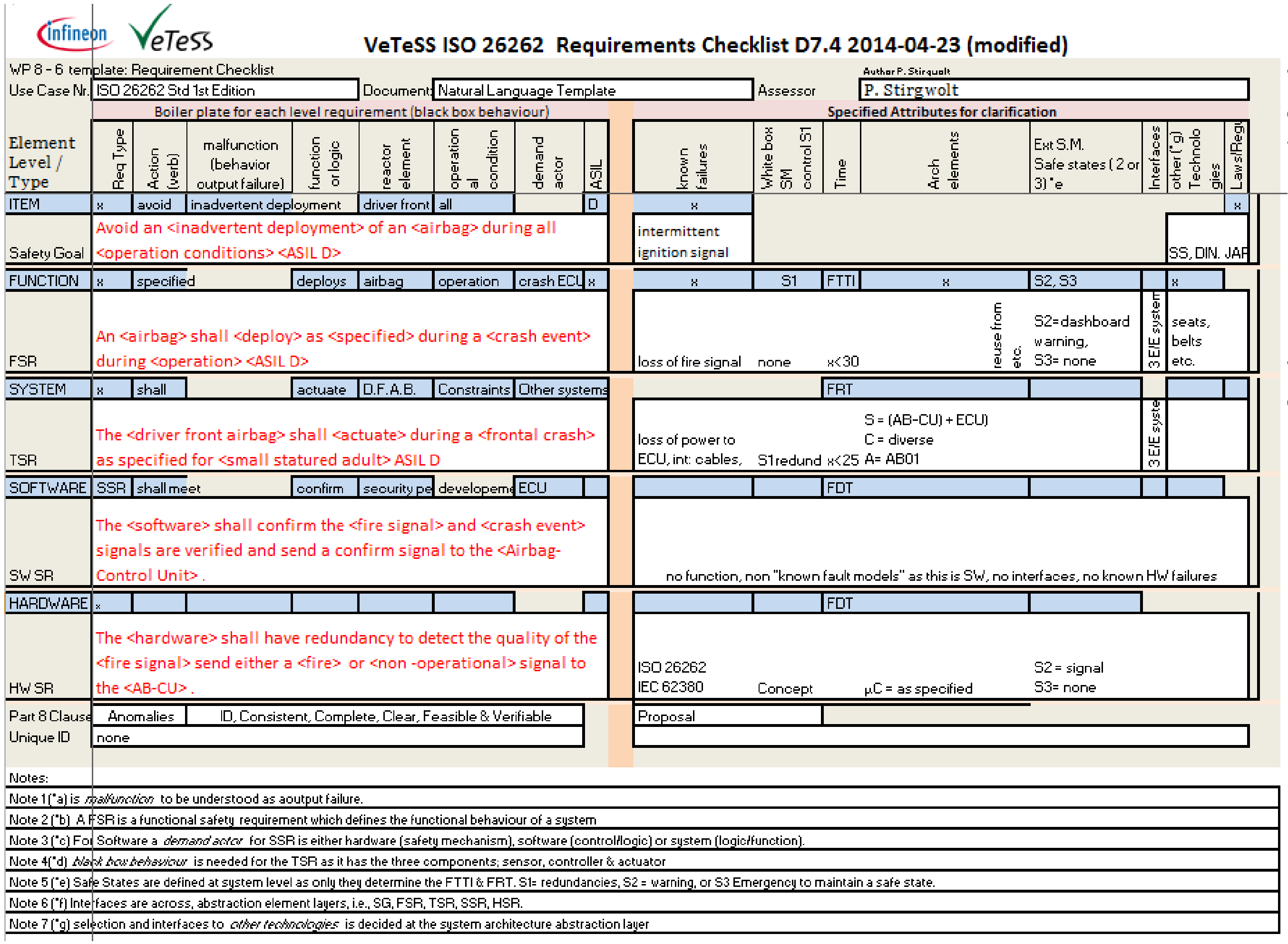
[14] “Systems Modeling Language (OMG SysML) - Version 1.3,” *OMG Standard*, 2012.

Definitions/Abbreviations

ASIL	Automotive Safety Integrity Level
bdd	Block Definition Diagram
E/E	Electric and Electronic
FSR	Functional Safety Requirement
HARA	Hazard analysis and risk assessment
HWSR	Hardware Safety Requirement
ibd	Internal Block Diagram
OEM	Original equipment manufacturer
SG	Safety Goal
SoC	System on Chip
SWSR	Software Safety Requirement
TSR	Technical Safety Requirement
uC	Microcontroller

Appendix

The ISO 26262 compliant requirements template is available as a spreadsheet, intended for the process of requirements capturing, either by single persons or teams of safety engineers.



VeTeSS ISO 26262 Requirements Checklist D7.4 2014-04-23 (modified)

WP 8 - 6 template: Requirement Checklist

Use Case Nr. ISO 26262 Std 1st Edition | Document: Natural Language Template | Assessor | Author P. Stirgwolt | P. Stirgwolt

Boiler plate for each level requirement (black box behaviour) | Specified Attributes for clarification

Element Level / Type	Req Type	Action (verb)	malfunction (behavior output failure)	function or logic	reactor element	operational condition	demand actor	ASIL		known failures	White box SM control S1	Time	Arch elements	Ext S.M. Safe states (2 or 3)*e	Interfaces	other (*g) Technologies	Laws/Regu
ITEM	x	avoid	inadvertent deployment		driver front	all		D		x							x
Safety Goal	Avoid an <inadvertent deployment> of an <airbag> during all <operation conditions> <ASIL D>									intermittent ignition signal							SS, DIN, JAR
FUNCTION	x	specified		deploys	airbag	operation	crash ECU	x		x	S1	FTTI	x	S2, S3		x	
FSR	An <airbag> shall <deploy> as <specified> during a <crash event> during <operation> <ASIL D>									loss of fire signal	none	x<30	reuse from etc.	S2=dashboard warning, S3= none	3 E/E system	seats, belts etc.	
SYSTEM	x	shall		actuate	D.F.A.B.	Constraints	Other systems					FRT					
TSR	The <driver front airbag> shall <actuate> during a <frontal crash> as specified for <small statured adult> ASIL D									loss of power to ECU, int: cables,	S1redund	x<25	S = (AB-CU) + ECU) C = diverse A= AB01		3 E/E syste		
SOFTWARE	SSR	shall meet		confirm	security pe	developme	ECU					FDT					
SW SR	The <software> shall confirm the <fire signal> and <crash event> signals are verified and send a confirm signal to the <Airbag-Control Unit> .									no function, non "known fault models" as this is SW, no interfaces, no known HW failures							
HARDWARE	x											FDT					
HW SR	The <hardware> shall have redundancy to detect the quality of the <fire signal> send either a <fire> or <non -operational> signal to the <AB-CU> .									ISO 26262 IEC 62380	Concept		µC = as specified	S2 = signal S3= none			
Part 8 Clause	Anomalies		ID, Consistent, Complete, Clear, Feasible & Verifiable							Proposal							
Unique ID	none																

Notes:

Note 1 (*a) is *malfunction* to be understood as aoutput failure.

Note 2 (*b) A FSR is a functional safety requirement which defines the functional behaviour of a system

Note 3 (*c) For Software a *demand actor* for SSR is either hardware (safety mechanism), software (control/logic) or system (logic/function).

Note 4 (*d) *black box behaviour* is needed for the TSR as it has the three components; sensor, controller & actuator

Note 5 (*e) Safe States are defined at system level as only they determine the FTTI & FRT. S1= redundancies, S2 = warning, or S3 Emergency to maintain a safe state.

Note 6 (*f) Interfaces are across, abstraction element layers, i.e., SG, FSR, TSR, SSR, HSR.

Note 7 (*g) selection and interfaces to *other technologies* is decided at the system architecture abstraction layer

Standard Compliant Co-Simulation Models for Verification of Automotive Embedded Systems

Martin Krammer, Helmut Martin, Zoran Radmilovic, Simon Erker, Michael Karner
{martin.krammer,helmut.martin,zoran.radmilovic,simon.erker,michael.karner}@v2c2.at
Virtual Vehicle Research Center
Graz, Austria

***Abstract*—The functional mockup interface (FMI) is a tool independent standard to support model exchange and co-simulation, as intended by the automotive industry to unify the exchange of simulation models between suppliers and OEMs. The standard defines functional mockup units (FMU) as components which implement the FMI. The creation and exchange of simulation models with customers and suppliers across the automotive supply chain is highly beneficial: In order to support early phases of development (requirement formulation, creation of executable specifications, and rapid prototyping) the creation of FMUs for co-simulation is reasonable. In this paper, we propose a structured method for generation of FMUs for co-simulation which are versatile, highly transportable and fast simulating. We show how to compile FMUs based on SystemC and SystemC-AMS, representing digital as well as analog and mixed signal electric and electronic systems. This tool-independent method allows inclusion of existing simulation models with only minimal adaptations. Additionally, no modifications of the standardized libraries are necessary with the outlined approach. The resulting FMUs allow convenient exchange and fast co-simulation of automotive systems, as they may be integrated by any FMI compatible master tool. An automotive battery system use case is shown to highlight these advantages and to demonstrate the simulation performance of the resulting FMUs.**

I. Introduction

Cooperative simulation, or co-simulation, has become a common method to support the development of automotive systems. The integration of different modelling languages, tools and solvers into one common co-simulation enables new possibilities for design and verification of complex systems. Efforts to standardize the exchange of simulation models and enable integration in co-simulation scenarios were undertaken by the ITEA2 MODELISAR project. One of its main goals was the development of the functional mock-up interface[1] (FMI) [1], [2]. The FMI is an open standard which defines an interface supporting model exchange between simulation tools and interconnection of simulation tools and environments. The second version of the FMI standard was released in 2014 [3], [4].

SystemC[2] [5] is a C++ based library for modelling and simulation purposes. It is intended for the development of complex electric and electronic systems. SystemC targets high abstraction level modelling for fast simulation [6]. It provides sets of macros and functions, and supports paradigms like synchronization, parallelisms, as well as inter-process-communications. Its simulation engine is included in the library, and is built into an executable during model compilation. While SystemC is capable of modelling and simulating digital systems, its SystemC-AMS[3] extension expands these concepts to the analog and mixed signal domain. Both, SystemC and SystemC-AMS libraries, provide a certain degree of protection of intellectual property, when optimized and compiled models are exchanged.

In this work, we present a tool-independent method on how to integrate electric and electronic system models together with their corresponding simulation engines into single functional mock-up units (FMU) implementing the FMI. Aforementioned models are built using SystemC and SystemC-AMS. By doing so, SystemC becomes available to a broad range of applications on system level in a standardized manner. The resulting FMUs are highly transportable and may easily be integrated within larger and more complex co-simulation scenarios for fast and convenient information exchange and system verification.

This paper is structured as follows. Section II recapitulates related work. Section III characterizes relevant frame conditions and requirements. Section IV introduces necessary steps on how to process models for FMU integration. Section V highlights the application of the proposed method with an automotive battery system use case. Section VI summarizes the results and concludes this paper.

II. Related Work

Since the release of the FMI standard version 1.0 in 2010 and version 2.0 in 2014, efforts have been spent in order to implement and test the functional mock-up interface, in order to build new workflows for simulation and verification of systems under development. This section presents related work in the area of FMI, FMU generation, as well as parsing and usage in simulation scenarios.
[7] introduces the FMI and argues about the necessity to share models for model/software/hardware-in-the-loop testing activities. As part of that a methodology for gradual integration and progressive validation is proposed. It also emphasises the need for conversion of existing models into the FMI standard.
[8] discusses technical issues and implementation of a generic interface to support the import of functional mock-up units into a simulator. For this import, the FMI calling sequence of interface functions from the standard are used.

[1] http://www.fmi-standard.org

[2] http://www.accellera.org

[3] http://www.systemc-ams.org

[9] describes the implementation of FMI in SimulationX. It presents code generation out of a simulation model for FMUs for model exchange and co-simulation. A code export step generates the necessary C-code for model exchange. For co-simulation, a solver is included in the resulting dynamic link library (DLL). The tool coupling using SimulationX is accomplished by using a wrapper.

The need for co-simulation in connection with the design of cyber-physical systems is highlighted in [10]. It follows the idea, that coded solvers in FMUs have some limitations regarding analysis or optimization. Therefore the authors strive for explicitly modelled ordinary differential equation solvers and claim a significant performance gain.

In [11] a verification environment using Simulink and SystemC is introduced. It relies on s-functions to create a wrapper in order to combine SystemC modules with Simulink.

In [12] the generation of FMUs from software specifications for cyber-physical systems is outlined. This approach fulfills the need for software simulation models. A UML based software specification is automatically translated into a FMU, maintaining its original intended semantics. This step is done using C-code, which is included within the FMU.

In [13] a high level approach for integration and management of simulation models for cyber-physical systems is shown.

[14] presents an integration strategy for rapid prototyping for Modelica models into the FMI standard, and highlights a high level approach for integration of cyber-physical systems.

SystemC and SystemC-AMS are used in a variety of simulation platforms, where different wrappers or adapters provide data exchange services. Examples thereof are given in [15], [16]. Regarding the use of SystemC or SystemC-AMS in the context of the FMI standard, no relevant publications are available to date, describing a unified process for integration. Thus, the outlined approach of integrating SystemC/SystemC-AMS models together with a functional mock-up interface into a FMU is considered as a novel contribution to the field of applied co-simulation.

III. Requirements on Modelling and Co-Simulation Data Exchange

In this section, general framework requirements and specifics are captured. This affects the SystemC and SystemC-AMS languages and libraries, the FMI standard, as well as co-simulation specific aspects. Usually, the compilation of SystemC or SystemC-AMS models leads to a platform specific executable, containing all models as well as necessary schedulers and solvers. This property seems suitable for application of SystemC's modelling and simulation concepts in context of the FMI. In order to comply to the FMI standard for co-simulation, a dynamic link library implementing the FMI needs to be compiled and assembled instead of an executable. The targeted FMI application scenario can be found in [17], [18] and is shown in Figure 1 for one single FMU. One co-simulation master is expected to coordinate the co-simulation by utilizing the FMI for communication to the FMU. The FMU includes the entire model and the corresponding scheduler or solver. SystemC and SystemC-AMS based approaches differ from co-simulation with tool coupling, as no separate tool is required for simulation execution.

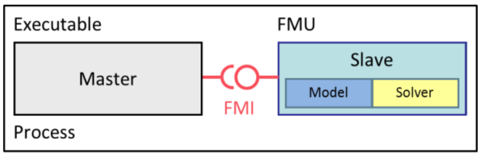


Fig. 1. Co-simulation with generated code on a single computer [18].

A state machine for the calling sequence of the FMI standard co-simulation interface C-functions is available [17, p.31]. For a basic repetition of simulation steps, the following principles can be distilled. First, a FMU is subject to instantiation and initialization. Then, input values, parameters and states are set using corresponding `fmiSet[...]()` group of functions. This is followed by a simulation execution phase called using `fmiDoStep()` function. If that step completes, the `fmiGet[...]()` group of functions is used to retrieve the results for external communication. In scope of this work the previously mentioned functions need to be implemented in SystemC and made available to the FMI based on a C-code interface. Since SystemC supports the concept of time, all C-interface function calls must be synchronized during co-simulation.

One further main criteria refers to the SystemC and SystemC-AMS libraries. Both libraries are available license free. SystemC is standardized under IEEE 1666 [5], whereas SystemC-AMS is documented in the Language Reference Manual version 2.0. Thus, it makes sense to strive for a solution which builds on these standard documents and does not cause any changes to the corresponding implementation libraries.

Under normal circumstances, SystemC and SystemC-AMS module simulations are performed in a single run, using the time domain simulation analysis mode. This means that the method `sc_start()` runs the initialization phase and subsequently the scheduler through to completion [19]. However, `sc_start()` may be called repeatedly with a time argument, where each simulation run starts where the previous run left off. For co-simulation, where data exchange and synchronization happens on discrete points in time, this function is vital to control simulation within the FMU. Memory management is rarely an issue in standard SystemC/SystemC-AMS models, due to their single elaboration phase and comparably short run times. In encapsulated FMUs, memory management can be crucial as the FMI standard suggests that FMUs have to free any allocated resources by themselves. The standard therefore defines instantiation and termination functions, therefore proper construction and destruction of all SystemC/SystemC-AMS models contained in FMUs is desirable to avoid any memory leaks.

The SystemC simulation kernel supports the concept of a delta cycle. One delta cycle consists of an evaluation phase followed by an update phase. This separation ensures deterministic behavior [19], as opposed to e.g. the use of events. Events trigger process executions, but their execution order within one single evaluation phase is non-deterministic. The concept of a delta cycle is even more important when moving from simulation level to co-simulation level, where external signals are connected to the model. New values written to e.g. signals become visible after the following delta cycle. Execution of one delta cycle does not consume simulated time.

SystemC and SystemC-AMS feature four different models of computation (MoC). According to [20], a MoC is defined by three properties. First, the model of time employed. Second, the supported methods of communication between concurrent processes. And third, the rules for process activation. SystemC features a kernel including a non-preemptive scheduler [5], which operates discrete event (DE) based for modelling concurrency. This MoC is used for modelling and simulation of digital software and hardware systems. SystemC-AMS features three different MoC, which may be used depending on the actual application. Namely these are timed dataflow (TDF), linear signal flow (LSF) and electrical linear networks (ELN). TDF operates on samples which are processed at a given rate, with a specified delay, at a given time step interval. TDF and DE MoC are synchronized using specified converter ports. LSF is primarily used for signal processing or control applications and features a broad range of predefined elements within the SystemC-AMS library. Converter modules for the conversion to and from the TDF and DE MoC exist. ELN permits the description of arbitrary linear networks and features an element library as well. Converter modules for the conversion to and from the TDF and DE MoC exist. Our goal is to support all four MoC in context of simulation through the FMI.

IV. Model Integration Method

In order to integrate and execute SystemC and SystemC-AMS simulation models in context of an FMU, we propose a structured method. The necessary steps are illustrated in Figure 2, indicated by the dashed box. They are described as follows.

(A) Modelling and simulation of a single component model.
(B) Model interface identification for coupling to the co-simulation environment.
(C) Wrapper class specification for controlling the model interface.
(D) C-interface specification for FMI integration.
(E) FMI integration using a predefined software developer kit.
(F) FMU compilation and assembly together with (architectural) model description.
(G) Integration of FMU to co-simulation master for simulation based system level verification.

Subsequently, each step is explained in detail.

A. Modelling and Simulation

To embed a SystemC or SystemC-AMS simulation model in a FMU, the model has to be set up and tested against its specifications first. Standalone executables may be executed, traced, and debugged using additional tools like an integrated development environment. However, a proprietary co-simulation master usually does not offer such sophisticated debug possibilities for compiled and assembled FMUs. To instantiate and test the model under development, a test bed is typically used. It may consist of stimuli generators, reference models or watchdogs [21]. The *SystemC Verification Library* (SCV) [22] is also available for this purpose.

B. Model Interface Identification

In a second step, the interface of the simulation model, which is later exposed through the FMI, shall be determined. This includes the definition of input and output quantities, or states, as well as internal timing (accuracy and precision required by the simulation model) and external timing (simulation step size for data exchange considerations. If a system level design is available, the model interface identification can be accomplished using these specifications.

C. Wrapper Class Specification

Typically, `sc_main()` is used to indicate the top-level module and to subseqently construct the entire module hierarchy through instantiation. The latter happens during the so called elaboration phase, right before the execution of `sc_start()`. For co-simulations, the breakdown of time into time steps is achieved by calling `sc_start()` multiple times. Thus it is necessary to keep the entire module hierarchy and its states persistently in memory, as seamless data exchange between simulation steps must be ensured. This can be achieved by using a C++ wrapper class defining a constructor and destructor managing the top-level SystemC simulation model in memory, until the entire co-simulation is finished. Listing 1 shows an example constructor for a SystemC wrapper class. Additionally, the wrapper class has `sc_signal` primitive channels attached. These are used for realization of the interface identified in step (B).

```
BatteryControllerWrapper::BatteryControllerWrapper() {
controller = new BatteryController("batteryController");

signalVoltageIn = new sc_signal<double>;
signalSocOut = new sc_signal<double>;
controller->in_voltage(*signalVoltageIn);
controller->out_soc(*signalSocOut);
}
```

Listing 1. Wrapper class constructor.

D. C-Interface Specification

The main idea here is to have a set of functions, which can be used to initialize, control, and finally shut down the C++-based SystemC/SystemC-AMS simulation model within a C-based FMU. To bridge the gap between the C-language FMI and the C++-language based simulation models, special linker instructions are required when compiling the file for the FMU. The compiler keyword used for this purpose is `extern "C"`. This C++ standard feature is a linkage-specification every compiler is required to fulfill. It exposes the enclosed C++ functions to the FMI.

For SystemC/SystemC-AMS, this means that signals (e.g. `sc_signal`) or ports (e.g. `sc_in`/`sc_out`) may be used for variable modifications when the scheduler or simulaton engine is not running. However, the scheduler requires the execution of one delta cycle to adopt a value which is written to a signal, otherwise the previous value remains assigned. This is achieved by using the `SC_ZERO_TIME` macro. It updates the signal's value while it does not advance simulation time.

For SystemC-AMS, the different MoC are combined advantageously using converters, in order to get and set input and parameter values as desired. To couple e.g.

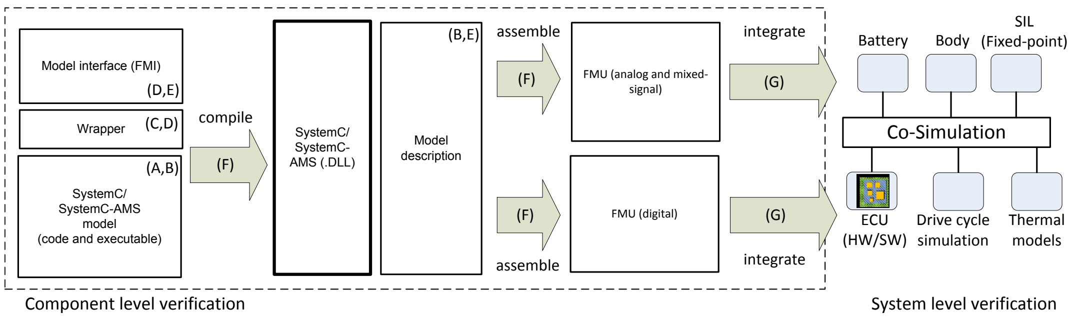


Fig. 2. The proposed process for the integration of executable simulation models into FMUs for co-simulation.

an electric current, the `sca_eln::sca_tdf_isink` and `sca_eln::sca_tdf::sca_r` primitive modules from the ELN MoC library may be used to read or write electric current values using the TDF MoC, respectively. A code example for setting a value to a TDF MoC module can be seen in Listing 2. Again, for execution of one delta cycle, the `SC_ZERO_TIME` macro is used.

```
extern "C" void setBatteryCurrent(double current) {
wrapper->batteryModule->cellGenerator->setCurrent(current);
sc_start(SC_ZERO_TIME);
}
```

Listing 2. Assignment of a value to a TDF module method

For FMI integration, a minimum of 6 different kinds of functions are required. `startInterface()` is required as an entry point to the SystemC/SystemC-AMS model, when simulation is initialized. Called once per FMU instantiation, this function calls `sc_main()` for the first time. One delta cycle is increased by calling `sc_start()` with the `SC_ZERO_TIME` argument. This triggers the construction of the simulation model in memory via the previously introduced wrapper class from step (C). This ensures that the simulation model is completely hierarchically constructed in memory and ready for simulation, without any simulated time passing by yet. `shutDownInterface()` is used to destruct all impressed models and free the occupied memory. `setValueXXX()` and `getValueXXX()` functions are used for each coupled variable to pass values through the FMI directly to the SystemC/SystemC-AMS model. The `doSim()` function is used to trigger the simulation start. It basically calls `sc_start()` with a SystemC time format parameter. If `sc_start()` is called using a fixed time interval, this time interval represents the step size of the FMU. In order to dynamically pass a required time interval setting to the simulation model, the `setTimeStep()` function is used. This realizes an adaptive step size co-simulation. The time step value is stored within the wrapper class. The call to `sc_start()` is modified accordingly, as seen in Listing 3.

```
sc_core::sc_start(wrapper->timeStep, sc_core::SC_SEC);
```

Listing 3. Dynamic simulation step size assignment.

E. FMI Intergration

For integration of the FMI, the FMU SDK (software development kit) is used as a basis [23]. It provides functions and macros which are included next to the SystemC/SystemC-AMS files. The main header file establishes a logical connection between the software code and the descriptive XML file that is required for each FMU. This `modelDescription.xml` file contains major information about the FMUs architecture. This also includes gathered information from step (B).

F. FMU Compilation and Assembly

Finally, the FMU parts are now compiled and assembled. According to [2, p.38], a FMU is referred to as a zip file with a predefined structure. The .dll file is placed in the binaries folder for the corresponding platform. The `modelDescription.xml` file is placed in the top level (root) folder. The SystemC/SystemC-AMS source files are placed in the sources folder optionally. Corresponding model documentation or associated requirements (see [24] for details) may be placed in the documentation folder. Initialization values, like sets of parameters, may be placed in the resources folder. After creating a zipped file from these contents, the FMU is ready for distribution and instantiation.

G. FMU Integration for Co-Simulation

The FMI for co-simulation standard defines a master software component which is responsible for data exchange between subsystems. In this work, the independent co-simulation framework (ICOS) from *Virtual Vehicle Research Center* [25] is used. Typically, it is applied to solve multidisciplinary challenges, primarily in the field of automotive engineering. Use cases include integrated safety simulation, electrical system simulation, battery simulation, thermal simulation, mechanical simulation, and vehicle dynamics simulation. It features an application design which separates the co-simulation framework and its coupling algorithms from the simulation tools that are part of the co-simulation environment. Hence the co-simulation framework is independent from the simulation tools it integrates [26]. The framework relies on the exchange of discrete time signal information, and provides several different algorithms e.g. for interpolation, extrapolation, and error correction methods [27] of these signals. The exchange of discrete time signals from one co-simulation component model to another is called coupling. The independent co-simulation framework implements the FMI standard and allows instantiation and co-simulation of FMUs. Alternatively, and to ensure FMI standard compliance, a *FMU checker* executable is provided with the standard. It instantiates an FMU and performs basic operations on it automatically.

V. Automotive Battery System Use Case

In order to demonstrate the functionality of the resulting FMUs described in Section IV, an automotive battery system use case was selected. The battery system introduced here is intended for a hybrid electric vehicle (HEV), where the battery powers an electric motor next to a combustion engine, as main components of the power train. Several strategies are known to charge and discharge an automotive battery propelling a vehicle. Information about driver behavior, routing, road profile, etc. have a strong influence on the behavior of the battery system.

Such a battery system usually consists of two main components, namely the battery pack as energy storage facility and a corresponding battery controller. The targeted battery pack consists of 4 battery modules, where each of them integrates 12 cells with a nominal capacity of 24Ah each. The targeted controller monitors operational condition and health of each of the modules. The controller also communicates with the car's hybrid control unit (HCU) to ensure stable vehicle operation. For the battery module, we construct a SystemC-AMS based FMU utilizing the ELN and TDF MoC. For the battery controller, we implement a SystemC/SystemC-AMS based FMU utilizing the DE and LSF MoC. Both FMUs are integrated into one common co-simulation scenario.

A. Battery Module FMU

The success of HEVs strongly depends on the development of battery technology. For the development of battery based systems it is essential to know the characteristics and the behavior of the battery. Especially in the field of functional safety, the knowledge of these features is key to control potential hazards. There is a wide range of different existing modes available for simulating the performance of lithium-ion battery cells. Most common approaches are electro-chemical models [28], [29], [30] and equivalent circuit models (ECM) [31], [32], [33], [34].

Electrochemical models describe the internal dynamics of a cell leading to complex partial differential equations with a large number of unknown parameters. Since these models are computationally expensive and therefore time consuming, they are not suitable for system-level modeling. ECMs on the other hand are computationally more efficient and are therefore better suitable for system-level modeling. Such models are also commonly used in embedded battery management systems (BMS) to estimate the state of charge (SoC) and predict the performance of physical batteries. An ECM is composed of primitive electrical components like e.g. resistors, capacities and voltage sources to simulate a cell's terminal voltage response to a desired current flow. It is capable to accurately describe the static as well as the dynamic behavior of a cell under various operating conditions. In [35] an ECM model of a battery cell intended for portable devices, written in SystemC-AMS, is shown.

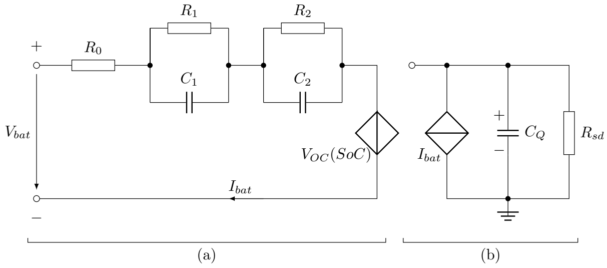


Fig. 3. Schematic diagram of the used cell equivalent circuit model: (a) Voltage-current characteristics, (b) Energy balance circuit.

For this use case, we model an ECM according to [34], [35]. The corresponding circuit diagram is shown in Figure 3. Such a two-RC block model (a) is a common choice for lithium-ion cells [32]. These two RC blocks characterize short-term and long-term dynamic voltage response of the cell, respectively, which arise from diffusion phenomena in the cell. I_{bat} in (b) represents an identified input current to the FMU according to section IV. The controlled voltage source V_{OC} reproduces the open circuit voltage (OCV), which represents an output of the FMU according to section IV. The serial resistor R_0 describes the internal resistance of the cell comprised of ohmic and charge transfer resistances and is connected in series with the two RC branches. In general, all parameters of the model depend on several quantities like state of charge (SoC), temperature, cell age as well as current direction and rate.

Mathematically the electrical behavior of an ECM with two RC branches can be expressed by Equations 1 and 2.

$$V_{bat} = V_{OC} + V_1 + V_2 + R_0 I_{bat} \tag{1}$$

$$\begin{bmatrix} \dot{V}_1 \\ \dot{V}_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{R_1 C_1} & 0 \\ 0 & \frac{1}{R_2 C_2} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} + \begin{bmatrix} \frac{1}{C_1} \\ \frac{1}{C_2} \end{bmatrix} I_{bat} \tag{2}$$

with $V_{1,2}$ and $\dot{V}_{1,2}$ the voltages across the RC branches and their time derivatives, respectively.

To parametrize an ECM, measurements on physical batteries are often performed to create large multidimensional look-up tables for the various parameters to cover all their dependencies. The lithium-ion battery model is based on data of a $LiFePO_4$ (LFP) battery [34]. To limit complexity for this use case we propose a simplified model, where the OCV depends on the SoC, $V_{OC} = f(SoC)$, and all other parameters are considered constant using averaged values from the data of [34]. The used $SoC - V_{OC}$ relationship is based on [36] and was adapted for usage out of its ordinary range to support simulation of e.g. overcharge scenarios. Figure 4 shows the $SoC - V_{OC}$ relationship. However, the model can straightforwardly be extended to a more advanced model by including additional elements representing terms for e.g. thermal behavior or aging effects. Thermal models may also be attached through co-simulation later on.

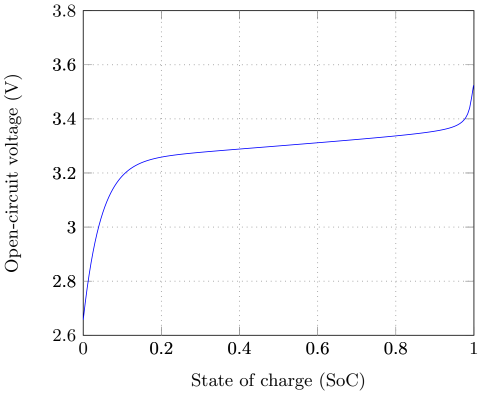


Fig. 4. Battery open circuit voltage V_{OC} as a function of SoC.

It is noteworthy that the ECM model can be adopted for other battery chemistries by simply adjusting the set of parameters used in the model. In context of a FMU, these could be placed within the FMU's resources folder in e.g. XML file format for model configuration during instantiation.

From a black box view, the proposed battery module FMU shall support the following functionality.

- One module shall consist of 12 cells in series connection, each modelled as described in this section.
- Current demand is an input to the battery module.
- Voltage is an output of the battery module.
- The battery module's SoC shall be set from external prior to simulation (without recompilation of the FMU).

A battery module with these properties is modelled in SystemC-AMS as follows. The ECM of a battery cell shown in Figure 3 representing equations 1, 2 is implemented using primitives of the ELN MoC of SystemC-AMS. The TDF MoC is used to interface the ECM, e.g. to control the voltage source V_{OC} the class `sca_tdf_vsource` primitive is used. The cell is instantiated 12 times, and these instances are connected in series to model a single battery module. The battery module is subject to integration into a FMU according to Section IV.

B. Battery Controller FMU

The electrochemical processes inside a battery are considered complex and the lithium-ion cell operating window is narrowed down by voltage and temperature restrictions. A battery controller monitors cells and modules to keep the battery pack and the entire vehicle functional within a safe state. Monitoring measures for batteries typically capture the state of charge (SoC), cell voltages as well as the temperature. The determination of the SoC is very complex, since it cannot be measured directly. Many vehicle applications require an exact knowledge of the SoC of the battery. The most obvious include the calculation of the vehicle's remaining driving range. More sophisticated applications include communications to the HCU, e.g. recuperation functions, influence on driving modes, trip routing, or comfort functions. Several solutions to the problem of accurately estimating the SoC have been proposed in literature [37]. The most common method for calculating the SoC is coulomb counting, which is based on measuring battery current. With the knowledge of an initial SoC_0, the remaining capacity in a cell can be calculated by integrating the current that is entering (charging) or leaving (discharging) the cell over time:

$$SoC = SoC_0 + \frac{1}{C_Q}\int_{t_0}^{t} \eta I_{bat}(\tau) \cdot d\tau \qquad (3)$$

Here C_Q is the rated capacity (the energy capacity of the battery under normal condition), I_{bat} is the battery current and η is a factor that accounts for loss reactions in the cells (we assume $\eta = 1$). Coulomb counting is straightforward to implement and able to determine the SoC under load, which makes it suitable for on-board applications. However, it requires the initial SoC of the battery. To get SoC_0 the cell voltage under no-load is measured and from this the SoC can be determined from the $SoC - V_{OC}$ relationship.

We propose a battery controller FMU, implementing four main functionalities:

- Sampling of the battery voltage.
- Sampling of the battery current.
- Calculation of the SoC based on $SoC - V_{OC}$ relationship and look-up table.
- Calculation of the SoC based on coulomb counting using current integration.

The battery controller is realized as a SystemC/SystemC-AMS module. It uses one thread for periodical voltage sampling and one for look-up table operations (based on OCV-V_{OC} relationship depicted in Figure 4). It utilizes the integrator primitive from the LSF MoC of SystemC-AMS for current flow calculation according to Equation 3. In this use case, no error correcting measures like a Kalman filter are implemented. However, in practice a solution is necessary to counteract drift effects. In the end, the module is compiled and assembled as an FMU according to Section IV.

C. Co-Simulation Integration

The resulting FMUs pass the *FMU checker* test and are integrated into the independent co-simulation framework as described in Section IV-G. First, its boundary condition server (BCS) is used to test the resulting FMUs. Second, the FMUs are integrated into a co-simulation scenario. This includes a mild hybrid vehicle together with a drive cycle modelled in CarMaker, and the hybrid controls modelled in Matlab Simulink.

D. Discussion of Results & Observations

The following Table I provides an overview of different simulation scenarios and their achieved performance within the independent co-simulation framework as described in Section IV-G. The simulated time is denoted as t_s, whereas t_e refers to the time needed for simulation execution on a standard laptop device.

Figure 5 shows the output of the controller's estimation of the SoC. In this scenario, the module is discharged with 20 A current pulses with a pulse period of $T = 1100\ s$ and a duty factor of $\tau/T = 1/11$. State of charge estimation with

Scenario	t_s [s]	t_e [s]
Battery controller w/ discharge pulses (BCS)	8000	9
Battery controller w/ drive cycle current (BCS)	200	<1
Battery module w/ discharge pulses (BCS)	8000	2
Battery module w/ drive cycle current (BCS)	200	<1
Battery module, controller w/ discharge pulses (BCS)	10000	15
Battery module, controller w/ discharge pulses (BCS)	20000	28
Mild hybrid vehicle drive cycle	200	17

TABLE I. SIMULATION TIME EVALUATION

a simple OCV-SoC lookup table does not make sense under load conditions, as during times $T - \tau$. Allowing a sufficiently large relaxation time after a discharge pulse we can compare the two methods and yield similar results for the SoC. Next, the controller and battery module FMUs are coupled to the vehicle model and its hybrid controls. Figure 6 shows an excerpt of the electric motor current demand and battery module state-of-charge as observed during a drive cycle.

From a qualitative point of view, the battery simulation results correspond to the results described in [36]. Quantitatively, the generated battery module FMU reproduces the simulation results shown in [35], validating against Li-Po cells from [8]. For this case no relevant increase of simulation time caused by the co-simulation framework was measured. The step size considered for co-simulation was 1s.

The ascertained overall efforts for FMU integration are considered justifiable, once the co-simulation interface has been defined. However, the following issues should be observed when following the proposed process. By encapsulating simulation models into FMUs, an additional layer of time synchronization is introduced. FMU-internal SystemC/SystemC-AMS module step sizes may take very small values and account for precise calculations. In contrast to that, a very small external co-simulation step size causes increased coupling-related communications, produces vast amounts of data, and therefore slows down simulation performance. From this it follows that the internal and external step sizes used may not diverge by higher orders of magnitude.

To ensure stable loading, execution, and unloading processes of FMUs at the FMI master, the use of pointers and dynamic memory allocation when constructing SystemC/SystemC-AMS modules is indispensable.

The FMI standard defines a resource folder inside a FMU, which may be used for e.g. different sets of ECM parameter settings. This is ideal for model exchange scenarios where models are kept separately from their associated parameters and configurations.

The creation of the required FMU XML file and synchronization to the model code causes additional efforts due to variable numbering and name assignments, especially if models are modified. Additional automation could help to improve these deficiencies, e.g. use of a model based software development approach.

VI. CONCLUSION

In this paper a structured method for the integration of SystemC/SystemC-AMS simulation models to the FMI standard is introduced. The presented method does not require any changes to the standardized SystemC or SystemC-AMS libraries. The method eases the integration of existing system

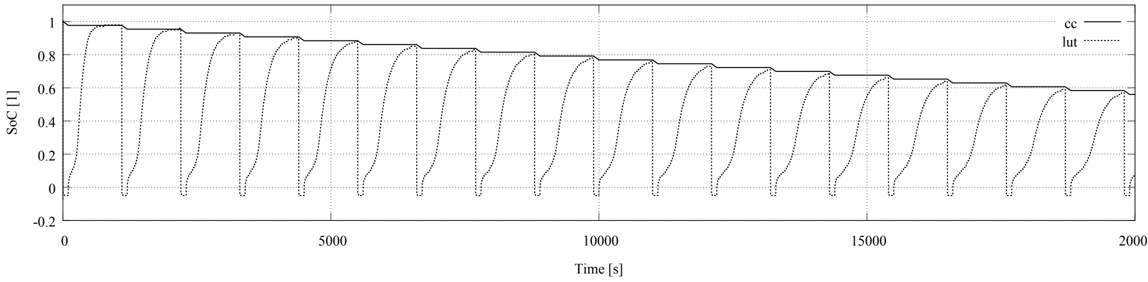


Fig. 5. Estimation of the state-of-charge using voltage based look-up table (lut) and coulomb counting with current integration (cc) approaches. For this simulation, a 20A pulse discharge test was conducted.

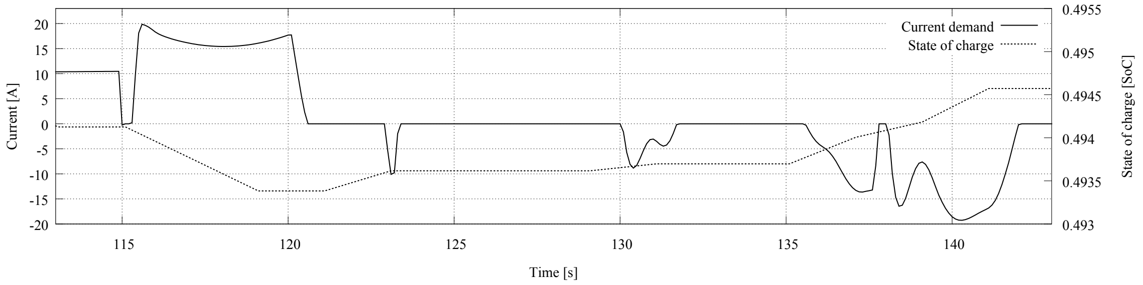


Fig. 6. The electric motor current demand (solid line) and battery module state-of-charge (dashed line) of a hybrid electric vehicle as observed during a drive cycle. The co-simulation scenario includes a vehicle power train model and a hybrid control unit model coupled to the two proposed FMUs.

level simulation models into larger and more complex simulation scenarios, which are used for information exchange and verification on system level. A two-part battery system use case from the automotive domain is presented, which exploits these MoC for simulation. The resulting FMUs created with the described method are highly transportable and configurable. These properties make them suitable for verification and information exchange processes within the automotive domain.

Acknowledgment

This research work has been funded by the European Commission within the EMC^2 project under the ARTEMIS Joint Undertaking under grant agreement n 621429. The authors acknowledge the financial support of the COMET K2 – Competence Centres for Excellent Technologies Programme of the Austrian Federal Ministry for Transport, Innovation and Technology (BMVIT), the Austrian Federal Ministry of Science, Research and Economy (BMWFW), the Austrian Research Promotion Agency (FFG), the Province of Styria and the Styrian Business Promotion Agency (SFG).

References

[1] T. Blochwitz, M. Otter, M. Arnold, C. Bausch, C. Clauß, H. Elmqvist, A. Junghanns, J. Mauss, M. Monteiro, T. Neidhold, D. Neumerkel, H. Olsson, J. V. Peetz, and S. Wolf, "The Functional Mockup Interface for Tool Independent Exchange of Simulation Models," *8th International Modelica Conference 2011*, pp. 173–184, 2009.

[2] "Functional Mock-up Interface for Co-Simulation, Version 1.0," 2010.

[3] T. Blochwitz, M. Otter, and J. Akesson, "Functional Mockup Interface 2.0: The Standard for Tool Independent Exchange of Simulation Models," *NAFEMS World Congress*, 2013.

[4] "Functional Mock-up Interface for Model Exchange and Co-Simulation, Version 2.0," 2014.

[5] "IEEE Standard 1666: SystemC Language Reference Manual," 2011.

[6] M. Barnasconi, "SystemC AMS Extensions: Solving the Need for Speed," *Design Automation Conference*, 2010.

[7] F. Corbier, S. Loembe, and B. Clark, "FMI technology for validation of embedded electronic systems," in *Embedded Real Time Software and Systems*, 2014.

[8] W. Chen, M. Huhn, and P. Fritzson, "A Generic FMU Interface for Modelica," *4th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools*, pp. 19–24, 2011.

[9] C. Noll and T. Blochwitz, "Implementation of Modelisar Functional Mock-up Interfaces in SimulationX," *8th International Modelica Conference 2011*, 2011.

[10] B. Pussig, J. Denil, P. De Meulenaere, and H. Vangheluwe, "Generation of functional mock-up units for co-simulation from simulink&reg;, using explicit computational semantics: Work in progress paper," in *Proceedings of the Symposium on Theory of Modeling & Simulation - DEVS Integrative*, ser. DEVS '14. San Diego, CA, USA: Society for Computer Simulation International, 2014, pp. 38:1–38:6.

[11] J. Boland, C. Thibeault, and Z. Zilic, "Using MATLAB and Simulink in a SystemC verification environment," *In Proceedings of Design and Verification Conference*, 2005.

[12] U. Pohlmann, W. Schäfer, H. Reddehase, J. Röckemann, and R. Wagner, "Generating Functional Mockup Units from Software Specifications," *Proceedings of the 9th International MODELICA Conference, September 3-5, 2012, Munich, Germany*, pp. 765–774, 2012.

[13] H. Neema, T. Bapty, and J. Batteh, "Model-Based Integration Platform for FMI Co-Simulation and Heterogeneous Simulations of Cyber-Physical Systems," in *Proceedings of the 10th International Modelica Conference*, 2014, pp. 235–245.

[14] A. Elsheikh, M. U. Awais, E. Widl, and P. Palensky, "Modelica-enabled rapid prototyping of cyber-physical energy systems via the functional mockup interface," *2013 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems, MSCPES 2013*, pp. 1–6, 2013.

[15] E. Armengaud, M. Karner, C. Steger, R. Weiß, M. Pistauer, and F. Pfister, "A cross domain co-simulation platform for the efficient analysis of mechatronic systems," *SAE World Conference*, no. SAE Technical Paper 2010-01-0239, pp. 1–14, 2010.

[16] M. Krammer, M. Karner, and A. Fuchs, "Semi-formal Modeling of Simulation-based V&V Methods to Enhance Safety," in *Proceedings of the Embedded World 2014 Exhibition and Conference*. Nuremberg, Germany: WEKA Fachmedien GmbH, 2014.

[17] "Functional Mock-up Interface for Co-Simulation, Version 1.0," 2010.

[18] "Functional Mock-up Interface for Model Exchange and Co-Simulation, Version 2.0," 2014.

[19] Doulos, *SystemC Golden Reference Guide*. Doulos, 2002.

[20] S. Swan, "An Introduction to System Level Modeling in SystemC 2.0," *Review Literature And Arts Of The Americas*, no. May, pp. 0–11, 2001.

[21] T. Grotker, *System Design with SystemC*. Norwell, MA, USA: Kluwer Academic Publishers, 2002.

[22] "SystemC Verification Standard Specification," 2003.

[23] Qtronic, "FMU SDK: Free development kit," 2014.

[24] M. Krammer, M. Karner, and A. Fuchs, "System design for enhanced forward-engineering possibilities of safety critical embedded systems," in *Design and Diagnostics of Electronic Circuits Systems, 17th International Symposium on*, April 2014, pp. 234–237.

[25] "ICOS Independent Co-Simulation - User Manual Version 3," Graz, Austria, 2013.

[26] W. Puntigam, "Coupled Simulation: Key for a Successful Energy Management," *Virtual Vehicle 11th Automotive Technology Conference*, 2007.

[27] M. Benedikt, D. Watzenig, J. Zehetner, and A. Hofer, "NEPCE - A Nearly Energy Preserving Coupling Element for Weak-coupled Problems and Co-simulation," in *IV International Conference on Computational Methods for Coupled Problems in Science and Engineering, Coupled Problems*, 2013.

[28] J. Newman and K. E. Thomas-Alyea, *Electrochemical Systems, 3rd Edition*. John Wiley & Sons, 2004.

[29] G. L. Plett, "Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs: Part 2. Modeling and identification," *Journal of Power Sources*, vol. 134, no. 2, pp. 262 – 276, 2004.

[30] M. Doyle, T. F. Fuller, and J. Newman, "Modeling of Galvanostatic Charge and Discharge of the Lithium/Polymer/Insertion Cell," *Journal of The Electrochemical Society*, vol. 140, no. 6, pp. 1526–1533, 1993.

[31] M. Chen and G. Rincon-Mora, "Accurate electrical battery model capable of predicting runtime and I-V performance," *Energy Conversion, IEEE Transactions on*, vol. 21, no. 2, pp. 504–511, June 2006.

[32] H. He, R. Xiong, H. Guo, and S. Li, "Comparison study on the battery models used for the energy management of batteries in electric vehicles," *Energy Conversion and Management*, vol. 64, no. 0, pp. 113 – 121, 2012, {IREC} 2011, The International Renewable Energy Congress.

[33] C. Birkl and D. A. Howey, "Model identification and parameter estimation for $LiFePO4$ batteries," in *Hybrid and Electric Vehicles Conference 2013 (HEVC 2013)*, Institution of Engineering and Technology. London: Institution of Engineering and Technology, 2013, p. 2.1–2.1.

[34] L. Lam, P. Bauer, and E. Kelder, "A practical circuit-based model for Li-ion battery cells in electric vehicle applications," in *Telecommunications Energy Conference (INTELEC), 2011 IEEE 33rd International*, Oct 2011, pp. 1–9.

[35] C. Unterrieder, M. Huemer, and S. Marsili, "SystemC-AMS-based design of a battery model for single and multi cell applications," in *Ph.D. Research in Microelectronics and Electronics (PRIME), 2012 8th Conference on*, June 2012, pp. 1–4.

[36] L. Lam, "A practical circuit-based model for state of health estimation of li-ion battery cells in electric vehicles," Ph.D. dissertation, TU Delft, Delft University of Technology, 2011.

[37] S. Piller, M. Perrin, and A. Jossen, "Methods for state-of-charge determination and their applications," *Journal of Power Sources*, vol. 96, no. 1, pp. 113 – 120, 2001, proceedings of the 22nd International Power Sources Symposium.

Systematic Pattern Approach for Safety and Security Co-Engineering in the Automotive Domain

T. Amorim[1], H. Martin[2], Z. Ma[3], Ch. Schmittner[3], D. Schneider[4], G. Macher[5], B. Winkler[2], M. Krammer[2], Ch. Kreiner[6]

[1] Technische Universität Berlin – Germany
buarquedeamorim@tu-berlin.de
[2] VIRTUAL VEHICLE Research Center – Austria
{helmut.martin, bernhard.winkler, martin.krammer}@v2c2.at
[3] Austrian Institute of Technology – Austria
{zhendong.ma, christoph.schmittner}@ait.ac.at
[4]Fraunhofer Institute for Experimental Software Engineering – Germany
daniel.schneider@iese.fraunhofer.de
[5]AVL List GmbH – Austria
georg.macher@avl.com
[6]Institute for Technical Informatics, Graz University of Technology – Austria
christian.kreiner@tugraz.at

Abstract — Future automotive systems will exhibit increased levels of automation as well as ever tighter integration with other vehicles, traffic infrastructure, and cloud services. From safety perspective, this can be perceived as boon or bane - it greatly increases complexity and uncertainty, but at the same time opens up new opportunities for realizing innovative safety functions. Moreover, cybersecurity becomes important as additional concern because attacks are now much more likely and severe. Unfortunately, there is lack of experience with security concerns in context of safety engineering in general and in automotive safety departments in particular. To remediate this problem, we propose a systematic pattern-based approach that interlinks safety and security patterns and provides guidance with respect to selection and combination of both types of patterns in context of system engineering. The application of a combined safety and security pattern engineering workflow is shown and demonstrated by an automotive use case scenario.

Keywords: ISO 26262 · SAE J3061 · Engineering Workflow · Safety Pattern · Security Pattern · Automotive

1 Introduction

Future applications in the automotive domain will be highly connected. They will rely on interacting functionalities exchanging data via various networking channels, and storing or receiving their operational data in or from the cloud. On the one hand, there

adfa, p. 1, 2011.

is enormous potential in these new types of cyber-physical-system (CPS) applications and services, which are bound to revolutionize the automotive domain, as we know it today. On the other hand, ensuring safety and security of next-generation automotive systems is a significant and comprehensive challenge that needs to be addressed before promising visions can become reality and an economic and societal success story.

Today, practitioners in the automotive domain are well experienced to deal with safety aspects during CPS development. However, there is a lack of knowledge on how to handle related security aspects, because the knowledge is either just non-existent or, maybe even more often, distributed over different organizational units in a company and thus not easily accessible.

Given the tight interconnection and the mutual impact of safety and security aspects, we argue that there is a need for a combined engineering approach enabling safety and security co-engineering. Moreover, given the present lack of experience in safety and security co-engineering, we think that providing additional guidance to engineers would be highly beneficial.

In this paper, we specifically focus on the proper and due consideration of the security aspect within a safety engineering lifecycle, which is one particularly urgent problem related to the aforementioned challenge. Consequently, we propose a systematic pattern-based and ISO 26262-oriented approach for safety and security co-engineering in the automotive domain. Through the use of patterns, we hope to close the security knowledge gap by harvesting its manifold benefits: conservation and reuse of design knowledge, best practices and tested solutions, reuse of architectural artifacts enabled by abstraction, cross-domain exchange of solution concepts, etc. Apart from the systematic interlinking of safety and security patterns, we elaborate how these patterns can be specified and maintained.

2 Background and Related Work

This section provides background knowledge about architectural patterns in general, safety patterns, security patterns, safety and security co-engineering, and current relevant automotive guidance for safety and cybersecurity.

2.1 Relevant Automotive Guidance for Safety and Cybersecurity

ISO 26262 – “Road Vehicles – Functional Safety” [1] is an automotive domain-specific safety standard. It provides a structured and generic approach for the complete safety lifecycle of an automotive E/E system including design, development, production, service processes, and decommissioning. ISO 26262 recommends requirements and techniques for system, software, and hardware design to achieve functional safety of E/E systems. For instance, the *Usage of established design patterns* is recommended (i.e. “+”) for all ASIL levels for each sub-phase of software development, as described in subsection 4.4.7 of Part 6. Concerning security, the first edition, released in 2011, does not consider it explicitly neither there is any support or guidance. The second edition, to be released mid-2018, is expected to provide some notes regarding the interaction of safety and security activities.

SAE J3061 [10] is a cybersecurity process framework for the development lifecycle of in-car systems. It provides guidance on best practice methods and techniques for secure system development tailored to the automotive domain by using a corresponding V model, as defined in ISO 26262. In J3061, safety and security interaction points are defined to coordinate the two engineering processes.

2.2 Safety and Security co-analysis and co-engineering

In our view, safety & security co-analysis refers to methods and techniques that can be used to identify safety hazards and security threats. Safety & security co-engineering refers to engineering activities that consider both safety and security and their interactions in the development lifecycle. Co-analysis includes activities in the early stage of the development lifecycle, e.g. in the requirements engineering as well as the design phase. Co-engineering considers all phases of the lifecycle, in which co-analysis is an integral part.

In the context of automotive domain, existing co-analysis methods Hazard Analysis and Risk Management (HARA) is standardized in ISO 26262 for safety, which can be extended with security Threat Analysis and Risk Assessment (TARA) method, as mentioned in SAE J3061 to identify cybersecurity risks [15]. Other proposals include Failure mode and Vulnerability Effect Analysis (FMVEA) [4] and Security Aware Hazard Analysis and Risk Assessment (SAHARA) [16] that aim at combining both safety and security analysis in parallel. A safety and security co-engineering approach should include all engineering activities in the automotive system development lifecycle according to relevant standards such as ISO 26262 and SAE J3061 based on the V-model [17].

2.3 Architectural Patterns

Patterns are used to solve similar problems with a general and universal solution. A well-known and proven solution for a specific problem is generalized so that it can be reused for similar recurring problems in other projects. Alexander describes the concept of using architecture patterns to solve similar problems in different projects [9].

The concept of patterns is used in many different domains including hardware and software. A good and very well-known reference is the book by Gamma et al. [11] (also known as the Gang of Four), which had a significant impact on making the pattern approach popular for software development. The book includes some general background and concepts as well as a collection of concrete patterns for object-oriented software design.

The state-of-the-art provides a few dozen safety architecture patterns [3][2], with some being just a variation of simpler ones. Armoush introduced in his PhD thesis [3] new safety patterns and provides a collection of existing safety patterns and a characterization of the main pattern representation attributes for embedded systems patterns (e.g. Name, Type, ID, Abstract, Context, Problem, Structure,…). These patterns are mostly based on the work of Douglas [12,13] for hardware patterns and on Pullum [14] for software fault tolerance techniques brought into pattern notation for software patterns.

Safety patterns usually include some kind of hardware redundancy, multiple channels with voters, or sanity checks [2]. They can address software or hardware issues and they allow systems to remain fully functional or to bring them to a safe state. Describing existing patterns, but the ones used in the presented case study, is out of the scope of this work.

Security engineering is an iterative and incremental process. Security patterns can be seen as the essence of sound security designs and best practices from an existing body of knowledge that can be used to solve security problems in new scenarios. During the security engineering process, security patterns can be used in requirements analysis and design to eliminate security flaws and provide additional information for security validation. Security patterns have attracted the attention of both academic researchers and industry [5]. The main focus of existing work is on the construction (including representation, classification, and organization) and application of security patterns. Security patterns are represented as textual templates or combined with UML models, in a hierarchically layered architecture or in a searchable pattern library. Security patterns have been proposed for requirements engineering, software system design such as web services, and Service-Oriented Architectures [6]. Open Security Architecture[1] is a community-based online repository of security control patterns based on the ISO 27000 information security standard family for enterprise IT systems, in which patterns are represented as text and graphical architecture designs in a consistent template. In recent years, security patterns have also been proposed for cyber-physical systems [7].

3 Methodology

Although patterns address specific problems, the context in which a pattern is applied influences how it should be applied. Therefore, more than a catalogue of patterns, practitioners require a workflow to systematically guide their efforts when using patterns to tackle safety and security problems. We propose a safety and security pattern engineering lifecycle that aims at combining the two engineering processes for pattern identification and design and allows for the necessary interaction and balancing of safety and security concerns.

3.1 Pattern Engineering Lifecycle

The Pattern Engineering Lifecycle is the approach proposed in this paper to help engineers selecting and applying safety and security patterns to develop safe and secure systems. The Pattern Engineering Lifecycle is meant to be used in unison (and tightly integrated) with the usual safety and security engineering approaches. It therefore does

[1] http://www.opensecurityarchitecture.org.

not substitute established approaches but rather enhances them with further tasks. The approach is suitable to be used with all existing patterns as well as ones to be developed.

The lifecycle takes place at the end of the *Product Development: System level* phase of the V-Model framework of ISO 26262 [1]. At this point, the *Functional* and *Technical Concept* are fully developed and both are used as input for the lifecycle. The output of the lifecycle is then consumed by the next phases of the V-Model, namely *Product Development: Hardware level* and *Software level*.

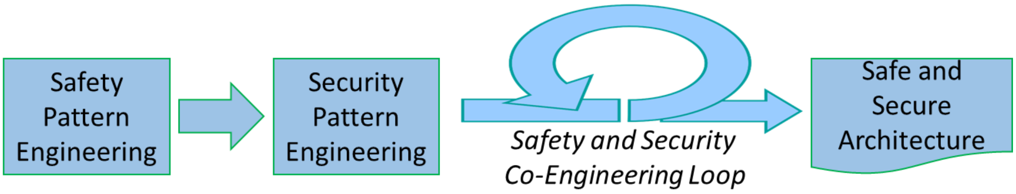

Safety Pattern Engineering
Security Pattern Engineering
Safety and Security Co-Engineering Loop
Safe and Secure Architecture

Fig. 1. Pattern Engineering Lifecycle

The lifecycle is divided into three main phases happening one after the other in a waterfall fashion (cf. Fig. 1). The first phase, Safety Pattern Engineering, comes before Security Pattern Engineering, the second phase. The rationale for this is that the approach explicitly focuses on "security for safety" (i.e., safety concerns are the main engineering drivers) and that security should start working when the final architecture is almost finished. Also, in general, further changes in the architecture might open new vulnerability points or might not be properly covered by mechanisms already implemented. However, security measures can influence system properties that can alter safety. For this reason, we introduce the Safety and Security Co-Engineering Loop, the third phase of the lifecycle. The loop prevents safety-motivated changes from creating unforeseen vulnerabilities and security-motivated changes from jeopardizing safety characteristics of the system. Each of these phases will be described in detail in the next paragraphs.

Safety Pattern Engineering

Safety Pattern Engineering involves safety-related tasks and is composed of three main tasks (cf. Fig. 2), which will be described in the following paragraphs.

Perform Safety Engineering

As described above, patterns are used to tackle specific problems; therefore, we need to have a good understanding of the system and the context in order to select and apply patterns appropriately. The workflow starts with established safety engineering approaches and techniques that need to be carried out until Safety Requirements (Functional or Technical) are available.

Select Safety Pattern

The decision about which pattern best fits a specific system should be analyzed taking into account the problem to be addressed and the context of the system. Besides, there are a few trade-offs that one needs to take into consideration when choosing an architectural pattern, such as costs (hardware, development effort) or standardization. The Safety Requirements guide safety engineers into selecting a safety. Current state-of-the-art [3][12][13] provides many patterns with detailed information about the impact in the system in the view of different dimensions (e.g. Cost, Reliability, Safety). There might be cases that no pattern is suitable for the discovered problems, thus the engineer needs to come up with an ad-hoc solution.

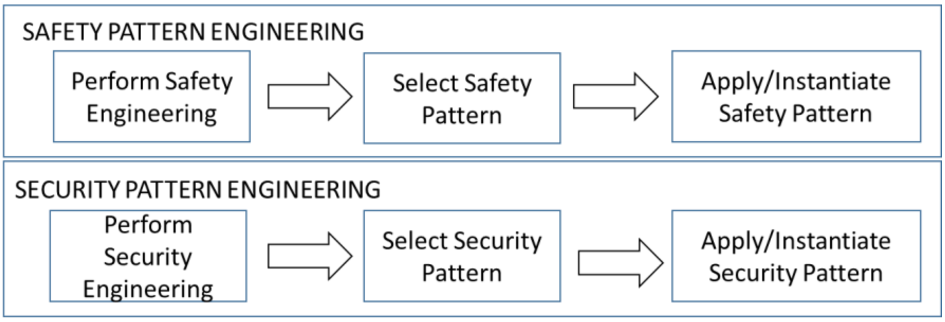

SAFETY PATTERN ENGINEERING
Perform Safety Engineering
Select Safety Pattern
Apply/Instantiate Safety Pattern
SECURITY PATTERN ENGINEERING
Perform Security Engineering
Select Security Pattern
Apply/Instantiate Security Pattern

Fig. 2. Safety Pattern Engineering and Security Pattern Engineering tasks

Apply/Instantiate Safety Pattern
The engineers should apply the safety pattern to the architecture, performing required changes on the architecture or on the pattern. Using the pattern “as-is” is usually not possible and some adaptation might be required. The updated system architecture is the prerequisite for the next task.

Security Pattern Engineering
In the previous phase, the architecture was updated with safety measures. In the second phase, Security Pattern Engineering, the architecture will be analyzed with regard to security vulnerabilities. The weak points are to be addressed by applicable security patterns and a secure architecture will be the output of this phase.

Perform Security Engineering
In this step, Security Engineering is performed on the existing system context such as functional requirements, results of Safety Engineering, and intermediate architectural design of the system, including the safety patterns. Established Security Engineering methods and techniques such as attack surface analysis, attack trees, and threat modeling can be used to identify vulnerabilities and threats. The results of this task leads to security measures that either mitigate potential threats or reduce the risks to an acceptable level. Special attention is given to vulnerabilities caused by safety patterns.

Select Security Pattern
The security engineers should give priority to the selection of re-usable security solutions from well-established security patterns for mitigating the security risks. If multiple security patterns are available, the selection of a security pattern is then a design

decision that optimizes cost-benefit. Similar to the selection of safety patterns, if no security pattern is available, an ad-hoc solution is applied.

Apply/Instantiate Security Pattern
In this step, the instantiated security pattern is incorporated into the existing system architecture design. If the information how to integrate is not available in the pattern description, the security engineers should adapt the security pattern to the specific system context and requirements.

Safety and Security Co-Engineering Loop
After the initial two phases of the Pattern Engineering Lifecycle, the Safety and Security Co-Engineering Loop starts. In this phase, lightweight versions of safety pattern engineering and security pattern engineering take place one after the other until no extra modification is required in the architecture. The fact that they are performed as a lightweight version means that the focus is on checking those aspects that experienced alteration and their respective influence on the overall system.

The Loop starts with the safety pattern engineering task requiring safety engineers to analyze how the newly added security patterns might impact the system safety. Some security architecture strategy might impair, for example, the communication time between components, causing a command to arrive late. Also in this task, the results of the first security pattern engineering phase help the safety engineers to identify further points of failure that could be caused by an attack. The initial safety pattern might require some modification to add extra safety.

On the other hand, if the newly proposed safety mechanisms imply new vulnerabilities or changes in the attack surface, the security engineers should detect, assess, and propose new solutions. This is what happens during the security pattern engineering performed in the lightweight version. This goes on like a cycle and stops when the system fulfills the desired safety and security requirements. Updating supporting documentation and updating the architecture are also tasks to be performed.

4 Implementation of Pattern Engineering Approach

In the following section, the technical implementation of the approach shall be demonstrated on an automotive case study.

4.1 Use Case Description

Our automotive use case example of a connected electrified hybrid powertrain is a combination of one or more electric motor(s) and a conventional internal combustion engine, which is currently the most common variant of hybrid powertrains. The variety of powertrain configuration options increases the complexity of the powertrain itself as well as the required control systems, which include software functions and electronic control units. With the integration of connectivity features, further novel vehicle functionalities and new business models can be discovered. Therefore, we focus on an integral part of every connected hybrid powertrain, the battery management system (BMS), and its functionalities related to the connection to the external world; in this case especially the connections with the charging unit.

In this paper, we investigate a specific use case scenario of the connected hybrid powertrain use case: charging of the battery system by connecting it with an external charging unit. Fig. 3 left shows the most relevant elements: battery satellite modules, battery management system, CAN communication, the charging interface, and the external charging unit.

4.2 Application of the approach

In this subsection, we apply the Pattern Engineering Lifecycle in the use case scenario presented in the previous subsection. The concept phase is considered in this example.

SAFETY PATTERN ENGINEERING

Perform Safety Engineering.
We describe in the following a small summary of the results of this task up to the level of Functional Safety Requirements:

Hazard: Wrong estimation of charging status.

Comment: The battery of electric vehicles can be very dangerous in case of overcharging, even causing explosions. If the charging status of a battery is estimated wrongly, extra energy might be supplied, leading to a hazardous situation.

Operational situation: Parking

Comment: The hazard will only happen while charging, and this can only be performed while the car is parked. This hazard might also occur while driving when architectures with regenerative systems are considered.

Hazard classification:

- Severity: 3 || Exposure of frequency: 4 || Controllability: 2
- Resulting hazard ASIL: [C]
- Safety goal: Estimate correct status of cycle while charging.
  - Safe state: Disconnect HV battery, Alert driver.
- Functional Safety requirement: Detect Failure and errors from BMS.

Select Safety Pattern
The results from Safety Engineering describe two possible safe states for the system that are compliant with the Safety goal. The “Disconnect HV battery” measure would cut off the power supply, the source of the hazard. The “Alert driver” measure would issue a warning to the driver. The car will be in parking mode if the hazard occurs (operational situation: Parking); therefore, full functionality in case of fault occurrence is not required.

We should apply to the architecture a pattern that helps fulfilling the Functional Safety Requirement “Detect Failure and errors from BMS”. We selected the Monitor-

Actuator Pattern [12] (cf. Fig. 3 Right) which provides heterogeneous redundancy. This pattern adds to the architecture a monitoring channel that detects possible faults and triggers the primary channel to enter its fail-safe state. The Monitor-Actuator Pattern is suitable to systems with low availability requirements and addresses the problem of finding an appropriate mechanism for detecting failures or errors without incurring higher costs.

Apply/Instantiate Safety Pattern
The Monitor-Actuator Pattern was instantiated as depicted in Fig. 3. Only changes to the BMS component were made.

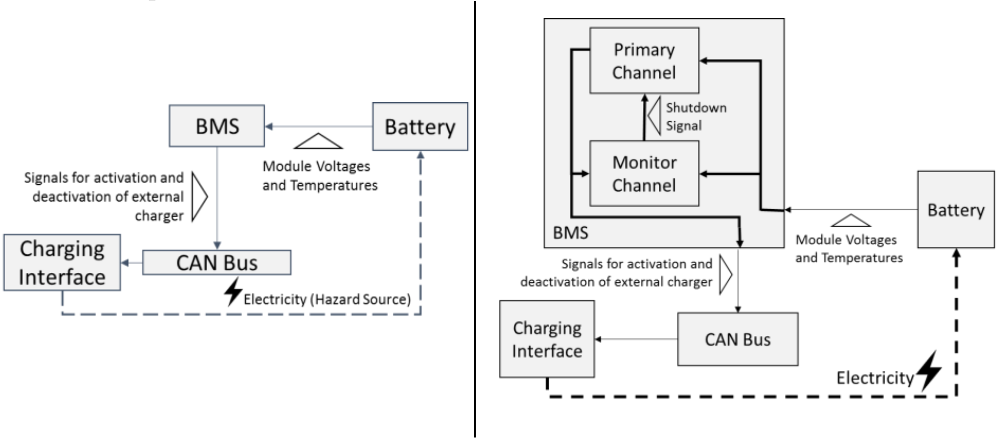

BMS
Battery
Module Voltages and Temperatures
Signals for activation and deactivation of external charger
Charging Interface
CAN Bus
Electricity (Hazard Source)
Primary Channel
Shutdown Signal
Monitor Channel
BMS
Battery
Module Voltages and Temperatures
Signals for activation and deactivation of external charger
Charging Interface
CAN Bus
Electricity

Fig. 3. Left: Automotive Battery Use Case | **Right**: Architecture with the safety pattern applied

SECURITY PATTERN ENGINEERING

Perform Security Engineering
In this context, Security Engineering follows the initial definition of a safety pattern to identify potential security vulnerabilities, threats, and risks in order to find appropriate countermeasures and apply corresponding security patterns. In this example, we use the threat modeling methodology [8], in which a system is modeled in a data flow diagram (DFD). When modeling the functional blocks from the safety pattern (cf. Fig. 3.) in a DFD, a few transitions and extrapolations occur. First, since threat modeling assumes that attacks happen when data flow from one process (i.e., a software component that takes input and either produces output or performs an action) to another, the logic signal flows in the safety pattern need to be translated into directional data flows according to the software architecture implementing this safety logic. Therefore, additional components are added such as the "CAN bus" process, which represents the communication bus in the in-car system. Second, the trust boundaries need to be defined in the DFD in order to identify attacks originating from data flows across trust boundaries. As a result, the charging interface is split into two parts: an in-car charging interface and the corresponding interface at the charging station. The interface on the charging station is modeled as an external interactor outside the "In-car system" trust boundary. There can be

different levels of trust boundaries. In this case, we assume that attacks can only originate from outside the "In-car system" boundary. Third, at the system level, security has an influence on components beyond the scope of the safety pattern. Since the communication between the primary and monitor channel and the charging interface goes through the CAN bus, and the powertrain unit is connected to the same bus, the security of the charging interface also influences the security of the powertrain unit. Thus even though the two safety modules cannot be attacked directly due to the unidirectional data flows, there are risks that an attacker might use the system charging function to attack the powertrain unit. Fig. 4 shows the modeled architecture in DFD using the Microsoft Threat Modeling Tool.

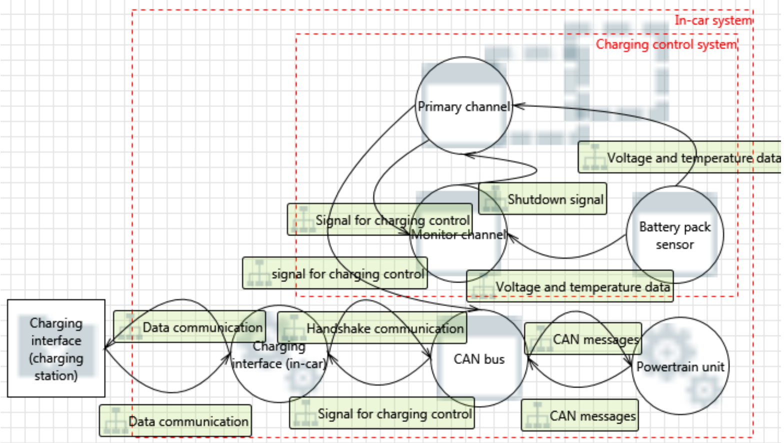


Fig. 4. Threat modeling of architecture (Tool: MS Threat Modeling Tool 2016)

The security analysis provides a list of threats according to the STRIDE method. In our case, the threats we identified are the communications from the external charging interface to the CAN bus that is responsible for establishing and maintaining communications for charging control. An attacker can use the in-car charging interface as an entry point by compromising the external charging interface or tampering with the communications between the interfaces to inject malicious content into the CAN bus.

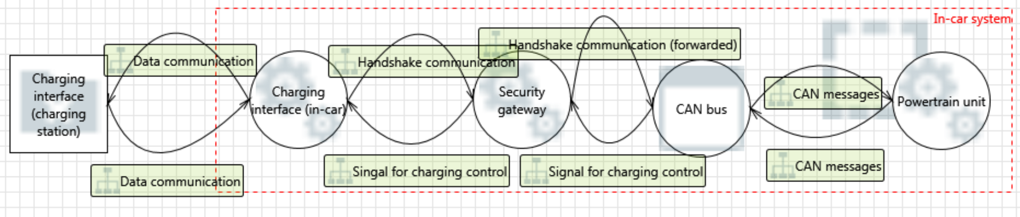

In-car system
Charging interface (charging station)
Data communication
Charging interface (in-car)
Handshake communication
Handshake communication (forwarded)
Security gateway
CAN bus
CAN messages
Powertrain unit
Data communication
Singal for charging control
Signal for charging control
CAN messages

Fig. 5. Security Gateway as a security pattern (Tool: MS Threat Modeling Tool 2016)

Select Security Pattern

One possible solution is to add a security gateway between the external unit and the internal CAN bus as shown in Fig. 5. The security gateway is a security pattern that is placed between an unprotected internal network and untrusted external entities when communication to the outside is inevitable. As a repeatable solution, the security gateway is not limited to the charging interface. It can be applied to any communication between the CAN bus and untrusted external devices. In general, it controls the network access to the internal ECUs according to predefined security policies and can also inspect packet content to detect intrusion attempts and anomalies. It can also serve as an endpoint for secure communication with external entities that implement network or application level securities. In this way, it adds security protection and segments the system without fundamentally changing the existing in-car system architecture.

Apply/Instantiate Security Pattern

In Fig. 5, we see the altered architecture with the Security Gateway module. Beyond the many benefits, a security gateway might introduce latency into the communication, which is a subject of safety impact analysis.

SAFETY AND SECURITY CO-ENGINEERING LOOP

First Safety Pattern Engineering Iteration

With the inputs from previous tasks we perform a HAZOP analysis to identify potential anomalies in the provision of the service controlling the Charging Interface (cf. Table 1). The focus is thus on the changes performed to the architecture by the security engineers.

Based on the analysis we identified failure modes Omission and Late as potential causes of a hazard (cf. Table 1). Other potential failure modes are not relevant for this scenario. As input from the Security Pattern Engineering phase, we get the information that the Security Gateway adds a small latency to the communication between the Charging Interface and the BMS. This small delay can cause a minor amount of extra charging in the battery which is not a source of hazard.

Table 1. HAZOP Guideword analysis of the architecture.

Function: Command to the Charging Interface to stop charging		
Guideword	**Possible Causes**	**Possible Consequences**
Commission	----	---
Omission	The Gateway blocks a message to stop charging. Message gets corrupted.	The Charging Interface keeps providing energy to the battery.
Early	---	---
Late	The extra processing time required slows the reaction time of the components.	Battery is charged for a couple of hundreds of milliseconds more than required.
Value High	---	---
Value Low	---	---

From the input received from the previous phase, we also discovered that the safety functions on the charging interface will not suffice in the case of a hacker attack. To tackle this issue a Charging Interface fail-safe device connected to the Monitor channel was integrated (cf. Fig. 6). Of course, one obvious drawback in this solution is the extra cost incurred due to extra hardware and installation.

First Security Pattern Engineering Iteration
The changes in the architecture neither create new vulnerabilities nor jeopardize the current mechanisms already in place. Since further modification of the architecture was not required, the Loop reaches an end. After finalization of the safety and security pattern engineering activities, the design can be reviewed to check whether all applied patterns can co-exist and whether there is no unwanted influence. While there is a direct review of the design with the applied patterns after each iteration, a final check can ensure the soundness of the design. It was decided to add the Security Gateway as an additional component in the system, to not only ensure that safety pattern and the security pattern do not interfere with each other, but also to support the maintainability of the security solution. Updates to the gateway do not impact the safety pattern directly.

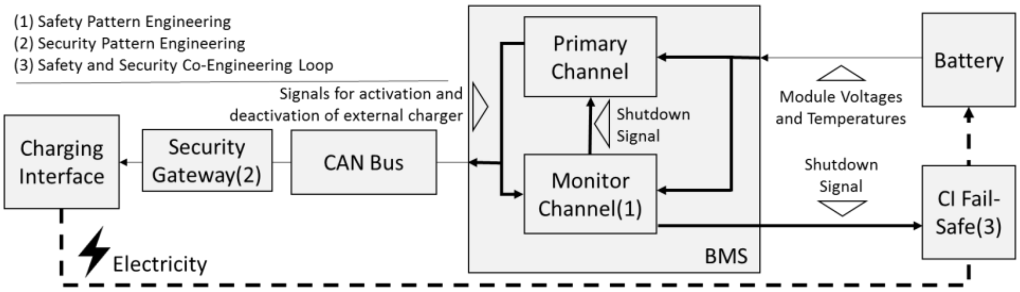

(1) Safety Pattern Engineering
(2) Security Pattern Engineering
(3) Safety and Security Co-Engineering Loop
Signals for activation and deactivation of external charger
Charging Interface
Security Gateway(2)
CAN Bus
Primary Channel
Shutdown Signal
Monitor Channel(1)
BMS
Module Voltages and Temperatures
Battery
Shutdown Signal
CI Fail-Safe(3)
Electricity

Fig. 6. Architecture after the first Iteration of Safety and Security Co-Engineering

5 Discussion

The availability of recurring process steps, based on automotive industry standards, results in faster and cheaper product development while fulfilling the need for intangible product properties, namely safety and security. This means that if, for instance, a safety (architectural) pattern is selected to address a specific safety requirement, additional information and guidance with respect to neuralgic aspects from a security point of view is needed. These might be subject to further security analyses and the application of an additional security (architectural) pattern might be warranted. The security pattern, in turn, can have a safety impact, which is again explicitly specified.

The decision about which pattern fits best for a specific system should be analyzed taking into account the problem to be addressed and the context of the system. Besides, there are a few trade-offs that one needs to take into consideration when choosing an architectural pattern, such as costs (e.g. available hardware, development effort) or standardization. These trade-offs are project specific can also involve managerial decisions.

As stated, safety and security engineering are very closely related disciplines and their synergy can be fostered when their similarities are recognized and adequate interactions are established correctly.

6 Conclusion and Future work

This paper focused on the selection, combination, and application of safety and security patterns. The introduction of the Pattern Engineering Lifecycle provided a systematic way of safety- and security-related pattern engineering process steps to development, and included already existing work products, such as the results of safety analyses. The Safety and Security Co-Engineering Loops helped to align these activities systematically. It benefits from tight integration of safety- and security-related process steps, which requires increased exchange of information between them.

An industrial use case demonstrated the practical realization of our approach: the architecture of an automotive battery system was described in a semi-formal way, including identification of its main components, physical interconnections, and flows of information. Within the Safety Pattern Engineering step, the “Monitor-Actuator Pattern” was selected as an appropriate measure for detecting failures originating from the BMS. Within the Security Pattern Engineering step, the “Security Gateway Pattern” was selected to protect the CAN bus from attacks on the Charging Interface. During the Safety and Security Co-Engineering Loop, the conducted HAZOP analysis identified additional modifications to the overall system. As result, a dedicated risk reduction measure was proposed to enhance the integrity due to combination of the two patterns. Finally, the complete system was presented after the first iteration of the introduced Safety and Security Co-Engineering Loop.

With the presented approach, we aimed to derive the manifold benefits from patterns inherent to their nature. This is a mean for accelerating the application of adequate safety and security co-engineering in the automotive domain. In particular, we showed a way to remediate the lack of security knowledge and facilitate easier and more informed integration of these two “separate” yet interfering disciplines. Future work should investigate an advanced model-based tool support for the proposed steps of the approach with interfaces to existing external tools.

7 Acknowledgment

This work is supported by the EU projects EMC² and AMASS. Research leading to these results has received funding from the EU ARTEMIS Joint Undertaking under grant agreement n° 621429 (project EMC2), EU ECSEL Joint Undertaking under grant agreement n° 692474 (project AMASS), and from the COMET K2 - Competence Centres for Excellent Technologies Programme of the Austrian Federal Ministry for Transport, Innovation and Technology (bmvit), the Austrian Federal Ministry of Science, Research and Economy (bmwfw), the Austrian Research Promotion Agency (FFG), the Province of Styria, and the Styrian Business Promotion Agency (SFG), the German Federal Ministry of Education and Research (BMBF), grant “CrESt, 01IS16043”.

8 References

1. International Organization for Standardization (2011) ISO 26262 - Road vehicles– Functional safety, Part 1–10. ISO/TC 22/SC 32 - Electrical and electronic components and general system aspects.
2. Preschern, C., Kajtazovic, N., & Kreiner, C. (2015) Building a safety architecture pattern system. In Proceedings of the 18th European Conference on Pattern Languages of Program (p. 17). ACM.
3. Armoush, A. (2010) Design patterns for safety-critical embedded systems (Doctoral dissertation, RWTH Aachen University).
4. Schmittner, C., Ma, Z., Schoitsch, E., Gruber, T. (2015) A case study of FMVEA and CHASSIS as safety and security co-analysis method for automotive cyber-physical systems. Proceedings of the 1st ACM Workshop on Cyber-Physical System Security. ACM.
5. Schumacher, M. (2003). Security engineering with patterns: origins, theoretical models, and new applications (Vol. 2754). Springer.
6. Delessy, N. A., & Fernandez, E. B. (2008, March). A pattern-driven security process for SOA applications. In Availability, Reliability and Security, 2008. ARES 08. Third International Conference on (pp. 416-421). IEEE.
7. Petroulakis, N. E., Spanoudakis, G., Askoxylakis, I. G., Miaoudakis, A., & Traganitis, A. (2015, December). A pattern-based approach for designing reliable cyber-physical systems. In Global Communications Conference (GLOBECOM), 2015 IEEE (pp. 1-6). IEEE.
8. Shostack, A. (2014). Threat modeling: Designing for security. John Wiley & Sons.
9. Alexander, C., Ishikawa, S., Silverstein, M., i Ramió, J. R., Jacobson, M., & Fiksdahl-King, I. (1977) A pattern language (pp. 311-314). Gustavo Gili.
10. SAE International (2016) J3061 - Cybersecurity Guidebook for Cyber-Physical Vehicle Systems
11. Vlissides, J., Helm, R., Johnson, R., & Gamma, E. (1995) Design patterns: Elements of reusable object-oriented software. Reading: Addison-Wesley, 49(120), 11.
12. Douglas, B. (2002) Real-Time Design Patterns: Robust Scalable Architecture for Real-Time Systems. Pearson.
13. Douglas, B. (2010) Design Patterns for Embedded Systems in C. Elsevier.
14. Pullum, L. L. (2001) Software fault tolerance techniques and implementation. Artech House, Inc., Norwood, MA, USA.
15. Macher, G., Armengaud, E., Kreiner, C., Brenner, Schmittner, C., Ma, Z., Martin, H., Krammer, M. (2017) Integration of Security in the Development Lifecycle of Dependable Automotive CPS, Handbook of research for cyber-physical systems Ubiquity, IGI Global.
16. Macher, G., Sporer, H., Berlach, R., Armengaud, E., & Kreiner, C. (2015) SAHARA: A security-aware hazard and risk analysis method. Design, Automation Test in Europe Conference Exhibition, (pp. 621-624).
17. Schmittner, C., Ma, Z., Gruber, T., Schoitsch, E., (2017) Safety and Security Co-engineering of Connected, Intelligent, and Automated Vehicles. ERCIM News #109.

Process- and Product-based Lines of Argument for Automotive Safety Cases

Helmut Martin, Martin Krammer, Robert Bramberger
VIRTUAL VEHICLE Research Center
Graz, Austria
{helmut.martin, martin.krammer,
robert.bramberger}@v2c2.at

Eric Armengaud
AVL List GmbH
Graz, Austria
eric.armengaud@avl.com

***Abstract*—The complexity of functions in today's vehicles demands a methodical procedure to ensure functional safety. Process audits and functional safety assessments confirm compliance to standards and safety of a product. One outcome of the safety life cycle is the safety case, which should "communicate a clear, comprehensive and defensible argument that a system is acceptably safe". In this paper, we propose a workflow to introduce a joint approach for process- and product- based argumentation compliant to ASPICE and ISO 26262. The approach is supported by argument patterns that cover the main lines of argument with respect to relevant standards. These patterns are elaborated in parallel to the development process and deal with visualization of the line of safety argumentation as well as the linking of evidences. They have a generic specification, provide templates and cover two argumentation aspects. Process-based argumentation deals with the engineering process and supports process audits whereas product-based argumentation deals with project specific outcomes, i.e. content of work products, and supports the functional safety assessment. The applicability of the approach is demonstrated on an automotive use case of a high voltage battery system for a hybrid electric vehicle powertrain.**

***Keywords* — ISO 26262, Automotive SPICE, Safety Case, Safety Argumentation, Safety Audit, Safety Assessment**

I. Introduction

The number of networked functions implemented on numerous control units in today's vehicles is increasing. The interaction of these heterogeneous functions causes a high degree of complexity which requires particular attention before, during and after development. From a safety point of view these functions must operate without any malfunctions, which could lead to hazards with catastrophic effects (e.g. harm people or lead to damage to the environment). The automotive safety standard ISO 26262 [1] defines an item as a system or array of systems, e.g. automotive Electric/Electronic (E/E) system, that implements a specific function. ISO 26262 provides requirements and recommendations concerning functional safety to handle required safety activities over the entire life cycle of an item, e.g. traceability over the elaborated work products. The standard compliance of the development process must be proven and the implemented product has to be safe according to recommended methods of the standard. The outcome of safety activities has to be documented in a multitude of work products. A work product is defined as a result, being associated with one or more requirements of ISO 26262. Furthermore, ISO 26262 demands conformation measures for relevant work products to check their correctness with respect to formality, content adequacy and completeness by an independent body or organization. In most cases, results are documents such as "Safety analysis", "Safety integrity determination" or "Reliability calculation". All documents as a whole provide evidence to compile the safety case.

To argue that all requirements concerning the process are fulfilled, adequate evidence is needed. However, evidence must clearly be distinguishable from the information, which led to it. From this point of view it is beneficial to find an improved methodology to indicate required evidence. The relationship between safety requirements and evidences has to be communicated by clear, comprehensive and defensible argumentation, to emphasize traceability.

All stakeholders, including engineers, reviewers and auditors, may not be in-depth familiar with the engineering process and the content of all resulting work products. Stakeholders will be able to comprehend the argumentation faster, resulting in shorter review cycles, concise feedback and a better understanding of the entire product development.

In section II the problem statement is formulated. Section III provides related work and the most important background information. Section IV describes the proposed methodology to use process- and product-based argumentation. Section V shows the application of the methodology in an automotive use case. Finally conclusions and future work are presented in section VI.

II. Problem Statement

Companies, which deal with safety critical products, engage external authorization bodies to certify their abilities concerning functional safety development (e.g. functional safety audit and functional safety assessment). Safety certification ensures that a certain product fulfills specific safety requirements in a specific environment. It requires a complete and structured collection of evidence to show that the developed system is acceptably safe.

The role of safety arguments is often neglected, thus stakeholders who are not directly involved in the creation of work products (e.g. reviewers) may have troubles to reconstruct the train of thought concerning decisions taken. Documentation of decisions in a comprehensible manner avoids loss of crucial information. A systematic approach is required to handle the development process that deals with dependency issues of the elaborated work products because the complex relationship between them may be not obvious. Artifacts cover outcomes of a specific engineering task, which include standard compliant work products. An argumentation method is needed that accompanies the process and is able to deal with the complex linkage between these individual artifacts. In order to come up with a versatile approach, being capable of dealing with a broad range of complex systems and processes, this method must be structured, modular and scalable.

For the identified problem, the following solution is proposed. Argumentation patterns and matching guidelines are defined and shall accompany the development process. They highlight the relationship between the development process and its related argumentation in an understandable way. A clear relationship between process- and product-based argumentation should be established in order to avoid systematic faults in the line of argumentation. A structured approach that offers a clear view on all present relationships will be easy to use and saves time and costs. The Goal Structuring Notation (GSN) [2] is defined for the construction of versatile arguments. In terms of ISO 26262 GSN helps to establish a valid relationship between evidence and safety requirements. Argumentation pattern should be elaborated to support corresponding artifact types. Process- and product-based argumentation is used together to compile a conclusive safety case.

III. Background and Related Work

A. Automotive Functional Safety - ISO 26262

ISO 26262 is the basis for development of safety-critical products in the automotive domain. It demands evidence to show that the established processes perform appropriately. ISO 26262 defines the "Automotive Safety Integrity Level" (ASIL) as a risk classification parameter for safety-critical hazardous situations. This is an important parameter and prescribes minimum efforts to be taken for all subsequent safety activities in the safety life cycle. The safety life cycle is defined, but ISO 26262 presupposes that special quality standards like Automotive SPICE [3] are fulfilled. An established quality level for processes is the basis for functional safety activities. Requirements in ISO 26262 expect various confirmation measures such as reviews (e.g. review of the safety analysis), audits (e.g. functional safety audit) and assessments (functional safety assessment). To pass these confirmation measures, it is beneficial if all necessary arguments are available without expenditure of time.

A very important topic in context of ISO 26262 is the elaboration of a safety case. It defines a safety case as "*the compilation of all work products that are used as evidence to show that all requirements for an item are satisfied. [...] The three principal elements are requirements, arguments and evidence*". Arguments explain the relationship between evidence and requirements (objectives). ISO 26262 does not provide detailed requirements concerning safety cases, even though distributed development is omnipresent in the automotive domain. ISO 26262 defines "Development Interface Agreements" (DIA) for clarification of the relationship between OEM and different suppliers (Tier x). DIA connects safety cases, if distributed development is performed.

If we have a look to other domains, it can be seen that safety cases are regarded as important and that they obtain a lot of attention. Depending on the context different stages of safety cases can be defined. The British "Office for Nuclear Regulation" [4] defines 11 principal stages in the life cycle of a nuclear facility. Kelly [7] defines three software safety cases based on the "MoD Defence Standard 00-55" [5] from the military domain.

In context of this paper the focus is on four stages which fit for automotive safety cases. They are explained in detail in section IV.A.

B. Quality Management - Automotive SPICE

Automotive SPICE is a quality development standard which is focused on improvement of development processes for software intensive systems. Automotive SPICE provides a process reference model which covers the entire product life cycle. The three belonging process categories are "Primary Life Cycle Processes", "Organizational Life Cycle Processes" and "Supporting Life Cycle Processes". They deal with all process aspects but functional safety aspects are only covered by referring relevant standards. A metric to assess process capability is part of Automotive SPICE. The quality of the process has to achieve at least the capability level "Managed process". With help of ISO 26262 safety and automotive related requirements are added to an Automotive SPICE compliant development process.

C. Process Line for Modeling of Process Elements

Safety-oriented Process Line (SoPL) [9], [10] defines a methodology which provides the opportunity to derive reusable standard compliant processes. The aim is to increase the number of reusable process elements. A process element is a representation of a specific standard compliant activity that includes roles, tasks, work products, tools and guidance. First relevant standards become analyzed and a standard compliant process model is build consisting of reusable process elements. The SoPL is able derive an executable project specific process tailored from a company specific process. The term „company specific" indicates that a pool of tools and methods has been defined to perform quality and safety related activities within the company. We use the SoPL as a basis for our work and extend this approach with safety argumentation methodology.

D. Modeling of Argumentation using GSN

GSN [2] is a graphical notation that can be used to document arguments. In GSN, an argument is defined as a series of connected claims. Strategy-elements are used to declare reasoning behind the connection between goals and sub-goals. Context-elements provide additional information to support a correct understanding of a specific argumentation part. Solutions are elements that support goals because they document pieces of evidence. The relationship between GSN elements is documented in a graphical way using different linkage elements (arrows). The two types of linkage elements are 'SupportedBy' and 'InContextOf'. The former, represented by lines with solid arrowheads, indicates inferential or evidential relationship, the later represented as lines with hollow arrowheads, declares contextual relationships.

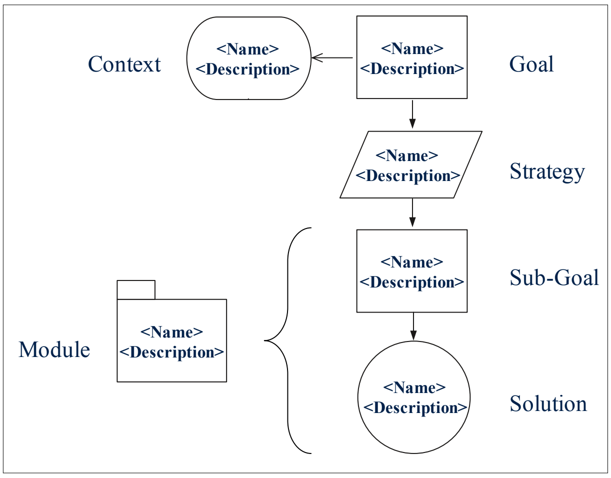


Fig. 1. Basic argumentation elements based on GSN standard

Modules are used to hide detailed structures and simplify goal structures to provide a general view. Between modules both types of linkage are possible. Furthermore, modules provide the opportunity to integrate argumentation from other sources if distributed development takes place and supports DIA of ISO 26262. Fig. 1. shows a simple goal structure for illustration. The angled brackets within elements represent metadata, in this case element <Name> and <Description>.

E. Related Argumentation Approaches

The following papers show a selection of different argumentation approaches, which investigate safety cases and argumentation topics. They emphasize the relevance to distinguish between process- and product-based argumentation in various ways.

The timely generation of well-focused safety cases is capable of bringing considerable benefit in the context of development and assessment and contributing to safety assurance of automotive E/E systems according ISO 26262. A process-based argumentation only renders the standard's implicit argumentation in a different form. Further argumentation is needed to provide a rationale argument for product-specific decisions during the development [13]. A process argumentation approach to generate process-based arguments from process models is shown in [11]. It reduces cost and time during certification process. Distinction between process- and product-based argumentation has been made in [14] but only product-based argumentation has been considered in detail. It deals with building of reusable safety cases and patterns.

The authors in [15] propose an integrated process- and product- based argumentation. Process-based arguments are backing arguments for product-based arguments to derive the safety case. The safety case development manual [17] provides guidance on the development of safety cases for the avionic domain. In this manual a clear distinction between product-based and process-based arguments is demanded since "the former is concerned with getting the right product and the latter with getting the product right."

IV. Methodology

The proposed methodology considers a company specific process which fulfills requirements from Automotive SPICE and ISO 26262. We show how to enhance the engineering process based on the SoPL approach by integration of safety argumentation modeling.

A. Definition of important Terms

To distinguish between process- and product-based argumentation, we introduce a categorization of work product as follows.

i.) Work products to prove capability and maturity of the development process (e.g. Project plan).

ii.) Work products to show compliance to ISO 26262. This type of artifact delivers proof that the defined process fulfils demanded safety aspects (e.g. confirmation review report).

iii.) Work products to ensure product safety. This type of artifact delivers product specific arguments which are needed in an assessment to show safety of the product (e.g. safety goals).

During the project life cycle, each of these work products goes through different stages of development. To ensure continuous argumentation throughout development, different states of work products must be considered. Therefore, the safety case should not be a final deliverable at the end of the project. To overcome this limitation, we introduce four stages of development based on [4] and [5] for the automotive safety case.

1. The "Preliminary Safety Case" is available after definition and review of the system requirements specification (functional safety concept is available).
2. The "Intermediate Safety Case" contains initial system design and preliminary validation activities. This type of safety case can be needed to get a permission to drive engineering prototype cars on public roads (cars are driven by professional drivers).
3. The "Pre-operational Safety Case" demonstrates that all necessary pre-operational actions have been completed, validated and implemented (basis for release for production).
4. The "Operational Safety Case" is available just prior to in-service use, including complete evidence of having satisfied the systems requirements (operational customer vehicle - field monitoring, maintenance).

Due to this variety of safety cases, it is necessary to have a systematic approach for their management. In this paper, we introduce such a systematic approach, which is applicable to all four stages. We focus on the first stage of the safety case exemplarily, the preliminary safety case.

B. Connect Standard Compliant Process and Argumentation

As shown in [10] a complete process takes aspects like functional safety and process quality into consideration. ISO 26262 formulates process requirements, which can be seen as a framework for tailoring. The direct derivation of a development process based on this single standard is not constructive. For the realization of an E/E system additional product specific standards have to be obeyed (e.g. EMC directive for hardware components, IEC 62660 for lithium-ion cells, etc.). Each supplier company determines its own priorities and way of engineering, and therefore defines its own specific development process. For that reason each process needs individual argumentation to prove standard compliance.

To ease the construction of an argument accompanying the development process, GSN argumentation patterns are used. A direct relation enables traceability between standards (e.g. ISO 26262, Automotive SPICE), process and argumentation (realized in GSN). This is important because traceability of arguments and requirements is a fundamental topic in current standards. If the arguments are provided in a systematic way, they are easy to comprehend and can be re-used by any stakeholder in a specific project. A good example thereof is field experience. In case of a cumulation of system failures in the field, the car manufacturer needs to take action. This likely requires engineers to comprehend design decisions which were made numerous product generations earlier.

Evidences in GSN argumentation structure are modeled as solution elements, which are directly process related. The name of a solution in development projects may differ from names used in the standard. For this reason work products designated standard compliant refer to outcomes created during process execution. The relation of product specific work products and standard compliant work products is given at any time.

C. Process- and Product-based Line of Argument

To deliver proof of functional safety for a defined development phase all requirements demanded by a standard (e.g. ISO 26262) have to be covered.

This section explains the difference between two types of argumentation, namely process-based and product-based argumentation. The proposed methodology defines each type of argumentation separately although they stay in direct relationship in the line of argument. Product development forces an established engineering process, supported by joint argumentation.

1) Process-based Argumentation

In case of process-based argumentation the arguments are directly associated with company specific processes which are derived from the Automotive SPICE process reference model as well as the ISO 26262 safety life cycle. Automotive SPICE contributes quality requirements which are presupposed by ISO 26262. During the process execution tools and methods which fit best are selected for a problem specific area of application. This selection leads to a project specific process. Process-based argumentation provides arguments to prove that the defined process fulfils demanded requirements. The argumentation is based on the existence of needed work products but not on their content. Usable work products are the types (i) and (ii) which have been defined in section IV.A. The approach in case of process-based argumentation is to document arguments, which support the process, in parallel with the process development. ISO 26262 demands functional safety audits to evaluate the implementation of the process and Automotive SPICE defines a quality assurance strategy to ensure the process quality. The process argumentation contains reasons why a particular process task has to be done in the described way. GSN elements like strategy and context are used to explain the decision why a goal splitting was done. Information about decisions is needed for process audits therefore it should always be documented. The GSN notation uses the possibility of unrestricted formulation to discriminate from the generic process and to emphasize arguments why the deviation is needed.

2) Product-based Argumentation

Within a generic formulated process a product specific decision determines a branch-off point. A decision based on a product specific requirement causes the necessity for different safety measures. For example, the development of a battery system requires different safety measures for battery packs with different capacity and different number of cells due to chemical and electrical issues. The safety measures are related to different software and hardware to manage the battery system. At that time the process becomes product-requirement-driven.

Product-based argumentation is elaborated based on content of available work products which are from type (iii) defined in section IV.A. With help of these work products it must be possible to establish an argument that the developed product is safe in terms of the relevant standards. Before project release for production a functional safety assessment has to be passed and arguments have to be prepared in a way that an external assessor can comprehend them. The focus of attention is to provide arguments why particular product related, technical decisions have been made and why specific methods or tools have been used.

D. Patterns - Development of reusable Artifacts

A pattern provides templates, guidance and formalisms to create goal structures for previously defined processes or products. The paper at hand uses definitions from [16] concerning patterns and templates and additional structural details of patterns which are defined in [6]. The most relevant attributes of patterns (based on [6]) are listed below:

- Intent of the pattern: What is the pattern for? (e.g. verification)
- Template: GSN argumentation structure used for implementation.
- Motivation: Scenario that supports the understanding (e.g. perform a complete verification)
- Applicability: Situations in which it can be applied (e.g. pattern is designed for HARA-verification in the automotive domain)

- Pitfalls: What possible pitfalls, hints or techniques should you beware of when using the pattern?
- Consequences: How does the pattern support its objectives? (e.g. pattern prevents users to make common mistakes)

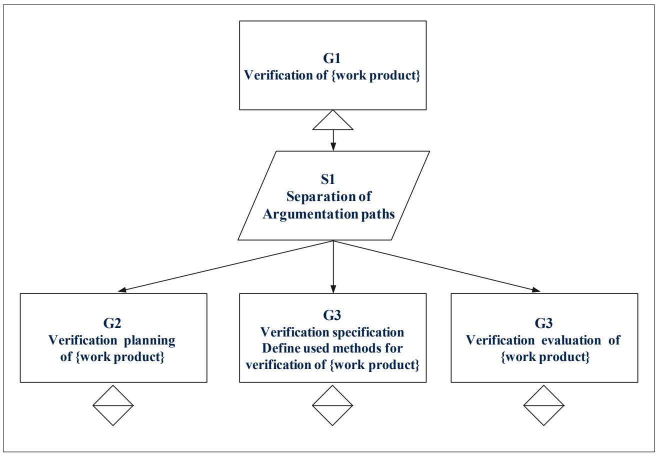

G1
Verification of {work product}
S1
Separation of Argumentation paths
G2
Verification planning of {work product}
G3
Verification specification Define used methods for verification of {work product}
G3
Verification evaluation of {work product}

Fig. 2. Template for verification activity

The objective of patterns is to support standard compliant safety argumentation and best practices from previous projects. Additionally it should be designed to be extendable and adaptable based on lessons learned. Patterns assist users by providing predefined elements, which are adaptable for tailoring needs. The pattern is not present in the final argumentation.

The focus of considerations is mainly on the attribute “Template” which contains a chain of arguments. Templates are graphical representations, i.e. argumentation structures, which contain symbols as well as text and require instantiation. Templates are basically reusable for similar lines of arguments. The GSN community standard provides two types of abstractions that are usable in templates, ”structural” and “entity”. Structural abstraction supports the concept of multiplicity and optionality and entity abstraction which provides the notions “Uninstantiated (UI)” and “Uninstantiated and Undeveloped (UU)”. A formal definition of these concepts is given in [8]. Graphical entities of GSN are annotated as uninstantiated, and may contain a textual expression in curly brackets to be replaced during instantiation. In templates the standard compliant name of a work product might be used as placeholder. The instantiation uses a project specific name. For illustration Fig. 2. shows a very generic template related to verification activities. Verification is split up to three activities which remain uninstantiated and undeveloped. The square at the bottom of the goals denotes that further development of the goals and instantiation of terms in curly brackets is needed.

A template is used twofold. The first aspect is related to decisions which are put into practice repeatedly whereby the line of argument is always identical. In this case instantiation is adding concrete project and evidence description. This use can often be found in connection with process argumentation (e.g. for the process to argue a HARA). The second use is when aspects of the product differ. In this case the provided templates have to be instantiated before they are applied (e.g. variation of product specific parameters).

E. Workflow for Introduction of Methodology

To introduce the proposed methodology to an engineering project, we define the following three sub-sequent phases which are shown in Fig. 3. .

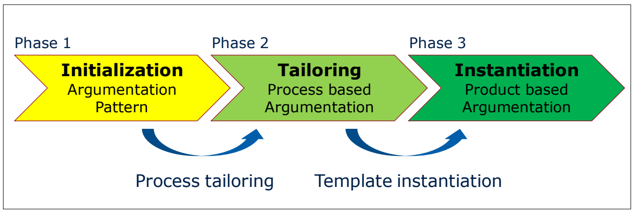


Fig. 3. Phases to create process- and product-based arguments

Phase 1 - Initialization of Development Process. The initialization phase is used to prepare all needed process elements to design a complete standard compliant development process. Activities in this phase are selection of relevant standards as well as identification of existing process and argumentation patterns which are suitable for reuse. The company specific process and the accompanying argumentation pattern are outcomes of this phase.

Phase 2 - Tailoring for process-based Argumentation. The tailoring from the company specific process to the project specific process means that process elements are selected to form the project specific development process. This selection includes the corresponding argumentation templates provided by patterns as well as methods and tools which should be used in the project. Creating a project specific process deals with decisions and judgments dependent on ASIL and needs expert knowledge. Process-based argumentation is needed for functional safety audits.

Templates are able to support process developers. They are used in two different cases. The first case provides arguments for repeatedly used generic process activities. This means, templates are available, which provide arguments that process requirements are fulfilled. In other words the template is included in the safety argumentation without changes. High level goals in a project are very similar to the company specific argumentation (e.g. the process for a HARA is quite similar in different projects). In the second case templates have to be instantiated because the project specific development process deals with activities beyond the template. This can occur if the process changes driven by a product or a customer demand. For example, one project uses HAZOP for hazard identification and in another project FMEA is required (see section V.B.) Fig. 4. shows an instantiated process template for HARA.

Phase 3 - Instantiation for product-based Argumentation. This phase covers product development by executing the project specific process. Templates for product-based argumentation support product specific decisions for a defined product. These decisions are made once and they are put into practice for a complete product line of battery systems. The generic template provides argumentation which is typically valid for battery systems (e.g. specific physical parameters like voltage or temperature). The complete argumentation structure is achieved by instantiation of the template to product specific context. The demand of a complete safety case is the main reason to elaborate product-based arguments for functional safety assessment. With help of results documented in work products it becomes easy to argue that product specific claims are valid. This argumentation is done bottom up starting with results of the development process. Furthermore, it is important to have quick access to related evidence that proves a product is safe.

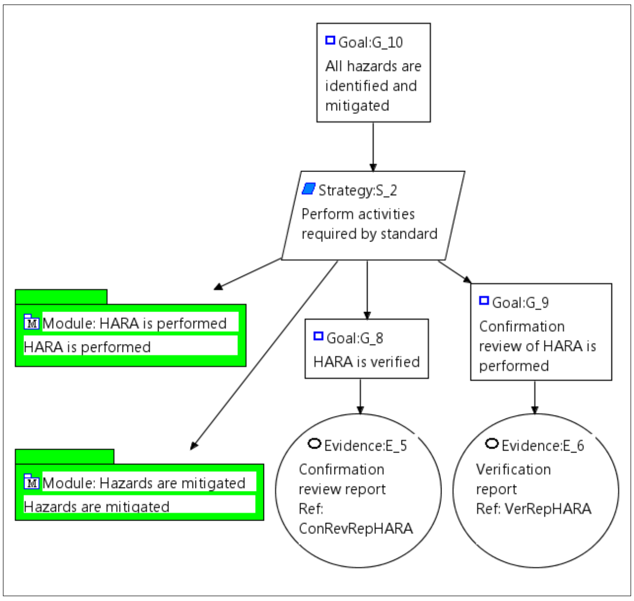

Goal:G_10
All hazards are identified and mitigated
Strategy:S_2
Perform activities required by standard
Module: HARA is performed
HARA is performed
Goal:G_8
HARA is verified
Goal:G_9
Confirmation review of HARA is performed
Module: Hazards are mitigated
Hazards are mitigated
Evidence:E_5
Confirmation review report
Ref: ConRevRepHARA
Evidence:E_6
Verification report
Ref: VerRepHARA

Fig. 4. Process-based argumentation for HARA (D-Case Editor)

V. Application to the Use Case

This section describes the application of the three phases (see Fig. 3.) in a concrete use case where argumentation modeling is implemented in the tool “D-Case Editor” [18].

A. Description of the Battery System Use Case

One major component of a Hybrid Electric Vehicle (HEV) powertrain is a High Voltage (HV) battery system. The current work focuses on the HV battery system for an automotive powertrain.

In the last decades, state-of-the-art technology is evolving and leading to the availability of various battery cell technologies with diverse characteristics (e.g. nickel-metal hydride, lithium-ion, and lithium polymer batteries). Some of the main targets for batteries for the HEV powertrain are low costs, high power density (e.g. >1200W/kg for HEV up to 250kW to support dynamic driving torques), very high cycle life time (e.g. >200.000 cycles of charge/discharge), high life time (e.g. >9 years), and safety. Safety becomes relevant because the power and energy density is increasing by decreasing of battery geometry, which leads to a potential increase of critical effects in the case of a critical malfunction [12].

The main functions of the battery system are providing electrical energy, storing/charging of electrical energy and electrical and thermal management. Based on these main functions potential safety-critical malfunctions can arise, e.g. overheating of battery cells, overcharging of battery cells and deep discharging of battery cells. These malfunctions could lead to following possible hazards: occurrence of high voltage, leakage of cell chemistry, toxic venting gas, fire and/or explosion.

Relevant data concerning the engineering process and safety aspects of the HV battery system was provided by the industrial project partner AVL. In the following section we show the first experimental results during the application of the proposed methodology.

B. Application of Workflow for Battery System Development

Application of the three subsequent phases defined in section IV is described in the following.

Phase 1- Initialization of Battery Development Process. The methodology is applied to develop a HV-battery system. ISO 26262 and Automotive SPICE are the standards which have to be considered in the regarded use case. The company specific process concerning the battery system is available. The argumentation patterns have been elaborated in parallel to the process. These patterns provide generic argumentation (e.g. HARA).

Phase 2 - Tailoring for Battery Development Process. The tailoring step derives the battery specific process and needed argumentation. Argumentation associated with the process is related to specific methods which have been selected (e.g. HAZOP for identification of hazards). The objective is to provide argumentation why HARA supports the goal that has to be achieved. Fig. 4. shows the argumentation concerning HARA starting with the goal “Hazards are identified and mitigated”. The list below shows the four argumentation paths required by ISO 26262 for a functional safety audit concerning HARA:

- HARA is performed
- Hazards are mitigated
- Verification of HARA is performed
- Confirmation review of HARA is performed

The audited engineering process is ready for execution to develop a HV battery system.

Phase 3 - Instantiation for Battery System Argumentation. Fig. 5. shows exemplary the product-based argumentation concerning overheating of a battery system for a hybrid car. In particular it deals with the argumentation related to a hazard which has been identified in a HARA. The hazard and situation analysis has been performed, hazardous events have been defined and classified by S/E/C parameters and the ASIL is determined. For the hazard “Overheating of the battery system” ASIL C has been determined for all charging situations of the battery system. The safety goal “Prevent overheating of the battery system” has been derived from this hazard. Related to this safety goal the safety measure “Temperature monitoring” has been defined.

In this example the safety measure is visualized as “Strategy” which is connected to sub-goals. The identified sub-goals lead to functional safety requirements which support the safety goal at the top. The functional safety requirements are stored in the project specific file “HV_Batt_FSR”. This file is linked to the ISO 26262 compliant work product “Functional safety concept”. It represents the product specific argument that implementation of derived requirements prevents the battery system of overheating. The file contains the evidence for a functional safety assessment.

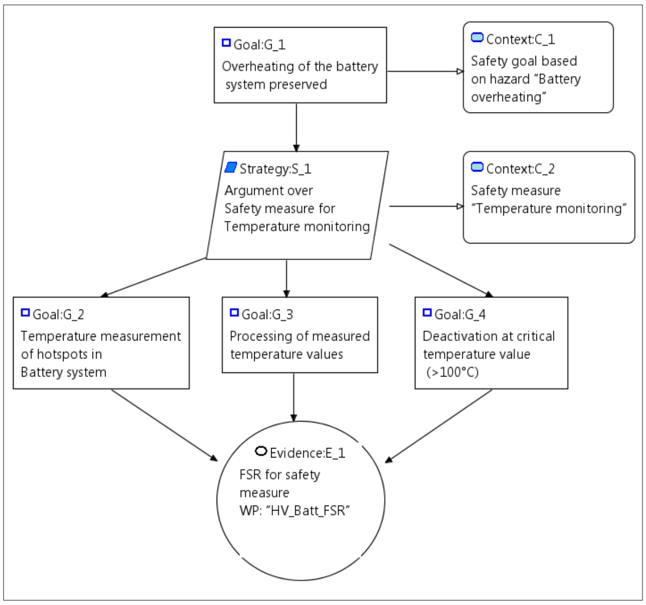

Goal:G_1
Overheating of the battery system preserved
Context:C_1
Safety goal based on hazard "Battery overheating"
Strategy:S_1
Argument over Safety measure for Temperature monitoring
Context:C_2
Safety measure "Temperature monitoring"
Goal:G_2
Temperature measurement of hotspots in Battery system
Goal:G_3
Processing of measured temperature values
Goal:G_4
Deactivation at critical temperature value (>100°C)
Evidence:E_1
FSR for safety measure
WP: "HV_Batt_FSR"

Fig. 5. Product-based argumentation for a battery system (D-Case Editor)

C. Evaluation Results

The application of the elaborated approach shows that the presented methodology is beneficial for safety case creation. Following benefits have been identified as results of evaluation.

- Argumentation patterns and the included structures accompany ISO 26262 and Automotive SPICE compliant processes.
- Traceability between argumentation goals and standard requirements is emphasized.
- Separation of process and product specific argumentation makes the methodology manageable and understandable.
- Reusable patterns and templates simplify argumentation and guarantee completeness.
- The elaborated argumentation structure reduces audit and assessment costs.
- The presented approach is usable for different stages of safety cases (see section IV.A).

VI. Conclusion and Future Work

This paper presents a methodology to create argumentation structures which are in direct relation to development processes and demanded requirements. The formalism deals with patterns and templates to make it easier to establish a complete understandable line of argumentation and prevents information loss. A workflow has been defined to introduce a methodology for process- and product-based argumentation. Project specific tailoring is used to create a standard compliant development process. Instantiation of templates provided by the engineering process leads to product specific safety argumentation. Application of the proposed workflow results in a complete and structured safety argumentation, which is needed for the safety case and supports functional safety audits and functional safety assessments. First experiences have been gained by successful application to an automotive battery use case to ensure compliance with ASPICE and ISO 26262.

As a next step, it is planned to evaluate tools that support GSN modeling. In cooperation with tool vendors of the EMC^2 project, existing tools will be enhanced to support the argumentation part of the presented methodology. In a long-term perspective it is intended to develop a tool which is able to support process development, process execution and process argumentation based on the proposed approach. A further aim is the extension of the methodology to cover security argumentation in a joint safety and security approach.

Acknowledgment

This work is supported by the EMC^2 project. Research leading to these results has received funding from the EU ARTEMIS Joint Undertaking under grant agreement n° 621429 (project EMC2) and from the COMET K2 - Competence Centres for Excellent Technologies Programme of the Austrian Federal Ministry for Transport, Innovation and Technology (bmvit), the Austrian Federal Ministry of Science, Research and Economy (bmwfw), the Austrian Research Promotion Agency (FFG), the Province of Styria and the Styrian Business Promotion Agency (SFG).

References

[1] International Organization for Standardization. “ISO 26262 - Road vehicles– Functional safety, Part 1–10.” ISO/TC 22/SC 32 Electrical and electronic components and general system aspects, Nov. 15, 2011.

[2] Goal Structuring Notation Working Group, GSN Community Standard Version 1, Nov. 16, 2011, www.goalstructuringnotation.info [Feb. 05, 2016]

[3] VDA QMC Working Group 13 / Automotive SIG, Automotive SPICE, Process Reference and Assessment Model, Version 3.0, 2015, www.automotivespice.com [Feb. 05, 2016]

[4] Office for Nuclear Regulation, An agency of HSE, The purpose, scope, and content of safety cases, NS-TAST-GD-051 Revision 3, 2013, Available: http://www.onr.org.uk/operational/tech asst_ guides/ns-tast-gd-051.pdf, [Feb. 05, 2016]

[5] U.K. Ministry of Defence, Defence Standard 00-55, Requirements for safety related software in defence equipment, 1997.

[6] T. Kelly, J. McDermid, Safety case construction and reuse using patterns, In: Daniel, P. (ed.) Safe Comp 1997, 1997, pp. 55–69.

[7] T. Kelly, I. Bate, J. McDermid, A. Burns, Building a preliminary safety case: an example from aerospace, Australian Workshop of

Industrial Experience with Safety Critical Systems, Sydney, Australia, 1997.

[8] E. W. Denney, G. J. Pai, A formal basis for safety case patterns, 32nd International Conference on Computer Safety, Reliability and Security (SAFECOMP 2013), LNCS 8153, Sep. 2013, pp. 21-32.

[9] B. Gallina, I. Sljivo, O. Jaradat, Towards a safety-oriented process line for enabling reuse in safety critical systems development and certification, In Post-proceedings of the 35th Software Engineering Workshop (SEW-35), Oct. 2012.

[10] B. Gallina, S. Kashiyarandi, H. Martin, R. Bramberger, Modeling a safety- and automotive-oriented process line to enable reuse and flexible process derivation, 8th IEEE International Workshop Quality-Oriented Reuse of Software, July 2014.

[11] B. Gallina, A model-driven safety certification method for process compliance, 2nd International Workshop on Assurance Cases for Software-intensive Systems, 2014.

[12] H. Martin, A. Leitner, B. Winkler, Holistic Safety Considerations for Automotive Battery Systems, *Automotive Battery Technology,* Springer International Publishing, 2014, pp. 1-17.

[13] Birch, John, et al., Safety cases and their role in ISO 26262 functional safety assessment, *Computer Safety, Reliability, and Security*. Springer Berlin Heidelberg, 2013, pp. 154-165.

[14] S. Wagner, B. Schätz, S. Puchner, P. Kock, A case study on safety cases in the automotive domain: Modules, patterns, and models, In Software Reliability Engineering (ISSRE), 2010 IEEE 21st International Symposium on 2010, pp. 269-278.

[15] I. Habli, I. Ibarra, R. Rivett, T. Kelly, Model-based assurance for justifying automotive functional safety, In Proc. SAE World Congress 2010, 2010.

[16] Agusta Westland Limited, BAE SYSTEMS, GE Aviation, General Dynamics United Kingdom Limited and SELEX Galileo Ltd., 2012, Available: https://www.amsderisc.com/wp-content/uploads/2013/01/MSSC_203_Issue_01_PD_2012_11_17.pdf, [Feb. 05, 2016]

[17] European Organisation for the Safety of Air Navigation, Safety case development manual, Edition 2.2, 2006.

[18] Y. Matsuno, D-Case Editor: A typed assurance case editor, In Proceedings of the 13th Real Time Linux Workshop, Prague, Czech Republic, 2011.

Safety and Security Co-Engineering and Argumentation Framework

H. Martin[1], R. Bramberger[1], C. Schmittner[2], Z. Ma[2], T. Gruber[2], A. Ruiz[3], G. Macher[4]

[1]VIRTUAL VEHICLE Research Center - Graz, Austria
{helmut.martin, robert.bramberger}@v2c2.at
[2]Austrian Institute of Technology - Vienna, Austria
{christoph.schmittner,zhendong.ma,thomas.gruber}@ait.ac.at
[3]TECNALIA / ICT Division - Derio, Spain
alejandra.ruiz@tecnalia.com
[4]AVL List GmbH - Graz, Austria
georg.macher@avl.com

Abstract. Automotive systems become increasingly complex due to their functional range and data exchange with the outside world. Until now, functional safety of such safety-critical electrical/electronic systems has been covered successfully. However, the data exchange requires interconnection across trusted boundaries of the vehicle. This leads to security issues like hacking and malicious attacks against interfaces, which could bring up new types of safety issues. Before mass-production of automotive systems, arguments supported by evidences are required regarding safety and security. Product engineering must be compliant to specific standards and must support arguments that the system is free of unreasonable risks.
This paper shows a safety and security co-engineering framework, which covers standard compliant process derivation and management, and supports product specific safety and security co-analysis. Furthermore, we investigate process- and product-related argumentation and apply the approach to an automotive use case regarding safety and security.

Keywords: Safety and security co-engineering • process- and product-based argumentation • process and argumentation patterns • automotive domain • ISO 26262 • SAE J3061

1 Introduction

The market and the society are requesting safe vehicles. Upcoming vehicle functions require external sensor data and communication across vehicle boundaries. Furthermore, software updates with new vehicle features can increase road safety, but these topics introduce the additional challenge on cybersecurity. Security issues are starting to be in the front line in the automotive business because more and more problems at the market occurred and have been published by various media. In 2015 the Jeep Cherokee become unfortunately famous for being hacked remotely [1]. Lately vulnerabilities in Tesla [2] have also become real. In both cases core safety-critical elements such as the brakes became vulnerable. The main lessons learned with these experiments are that vulnerabilities are hidden in the inner design of the system. Security has to be considered at early stages of the concept design [3].

The industry and standardization committees are moving forward a collaborative approach between safety and security disciplines. Currently, automotive safety and security disciplines are not similarly mature - security is less mature than safety [4]: While the SAE guidebook regarding automotive cybersecurity is available in the first edition, for the established automotive functional safety standard ISO 26262 [5] the preparation of edition 2 is ongoing. Both documents note interaction points of functional safety and cybersecurity[1], but only in an informative way. The standards focus on guidance to solve the challenges in the specific safety and security lifecycle. One of the challenges identified in the ISO 26262 standard is the need of a safety case which provides argumentation in a clear and comprehensive way that a system achieves a reasonable level of functional safety to operate in a given context. While functional safety refers to safety against failures in electrical/electronic (E/E) components, in the future there has to be argumentation where not only safety but also security and probably other dependability aspects are covered.

The paper at hand deals with a concept that covers standard compliant process- and product-based argumentation in context of safety and security. Just by following the standards procedures, automotive systems are not guaranteed to be free of risks. Standards are considered a compilation of best practices which describe industry-wide accepted concepts, methods and processes. The paper is structured as follows: Section 2 describes the state of the art and previous approaches for this problem. Section 3 presents the safety and security co-engineering framework proposed by the authors. Section 4 demonstrates how the approach is put into practice by using specific tools. Section 5 provides conclusions and an outlook on further work.

2 Background and Related Work

ISO 26262 is the automotive functional safety standard, describing a safety lifecycle for the development of safety-related automotive systems (targeting passenger cars and minivans). The first edition was published in 2011 and is currently in a revision phase. A new informative annex will define potential interaction and communication channels between functional safety and cybersecurity. The same concept of safety and cybersecurity interaction points is presented in SAE J3061 [6]. The security lifecycle specified in SAE J3061 proposes communication paths between safety and security engineering. Fig. 1 provides an exemplary overview of the interaction between safety and security engineering during the concept phase. The lifecycles itself are clearly described in the standards, but the interaction and cooperation is currently based on informative annexes which suggest approaches and potential cooperation topics.

[1] The term “safety” refers functional safety according to ISO 26262, and “security” refers to cybersecurity according to SAE J3061.

There is a need to define activities to force interaction between the standards. Based on SAE J3061 a joint working group between ISO and SAE was started with the goal of developing an SAE/ISO "Standard for Automotive Cybersecurity". For safety and security co-analysis in different lifecycle phases multiple methods have been developed, e.g. STAMP (Systems-Theoretic Accident Model and Processes) [7] a theoretic model for safety, SAHARA (Security-Aware HARA) [8], an extension of the HARA method (Hazard And Risk Analysis) or FMVEA (Failure Modes, Vulnerabilities and Effects Analysis) [9], a combination of threat modeling and failure modes and effects analysis. But methods like these need to be embedded in a larger lifecycle framework.

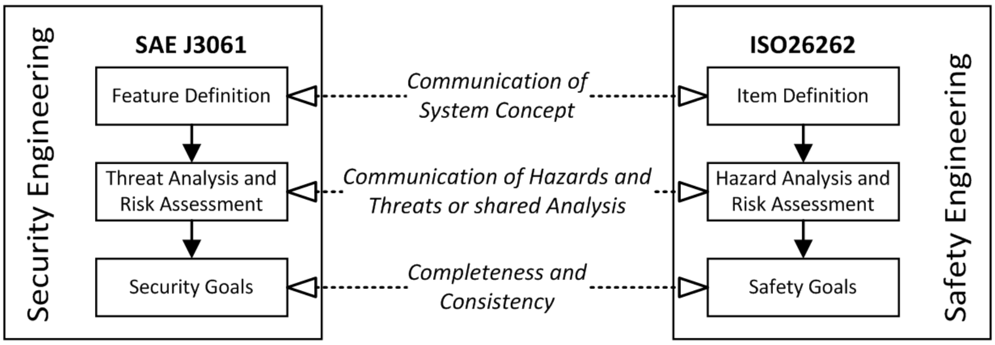

Security Engineering
SAE J3061
Feature Definition
Threat Analysis and Risk Assessment
Security Goals
Communication of System Concept
Communication of Hazards and Threats or shared Analysis
Completeness and Consistency
ISO26262
Item Definition
Hazard Analysis and Risk Assessment
Safety Goals
Safety Engineering

Fig. 1. Comparison of safety- and security engineering

For safety and security it is required to provide evidence and argumentation to show that system development was done compliant to relevant standards and that the system satisfies safety and cybersecurity goals. The final documentation has to be provided by the assurance case including safety and cybersecurity.

ISO 26262 mentions the possibility to use a graphical notation **Goal Structuring Notation (GSN)** to create the safety case. GSN's initial intention was to support safety case management [10]. Ray and Cleaveland proposed to apply GSN for constructing security assurance cases of medical cyber-physical systems [11]. The graphic structure of the security assurance case starts with a top-level security claim node accompanied by context information node and then breaks into layers of sub claim nodes that argue over different stages and aspects of the development lifecycle. Each sub claim is supported by a set of evidence nodes that explain the validity of the claim. Basically, GSN for assurance case is a graphic way to organize narrative information of claim, context, strategy, argument and evidence according to the GSN convention.

Patterns assist in reusing best practices systematically [17]. They are a suitable way to support argumentation that safety and security related requirements are fulfilled. Menon et al. [12] demonstrate how patterns are used to provide argumentation structures for software safety arguments. The authors define the structure consisting of GSN elements and its applicability. Patterns are usable on all development levels. Preschern et al. [13] examine the relationship between security and functional safety. The authors present an approach to categorize threats related to the impact to safety-critical functions. Taguchi et al. [14] define and compare different types of patterns concerning safety and security.

3 Safety and Security Co-Engineering Framework

¡Error! No se encuentra el origen de la referencia. shows the main steps of the proposed methodology which considers all process steps necessary in an automotive safety and security related development project:

Regulations and Standards (I) and Process Definition (II). In a first step we identify all relevant regulations and standards. In our automotive use case we deal with ISO 26262 regarding road vehicles functional safety and SAE J3061. It is challenging to match these two topics because they are influencing each other. Process definition has to consider that elaborated process steps are not only in parallel but also highly interactive, especially when we have to handle functional safety and cybersecurity. In addition, processes have to incorporate special analysis methods, which handle safety and security aspects in one common analysis methodology. Integrated processes which are basis for co-engineering unite safety with security activities. They lead to integrated requirements, work products and argumentation.

Process Management (III). The core of the framework is the distinction between functional safety and security related process and product requirements and the identification of interactions. Process requirements describe activities and steps, which are demanded by standards, while product requirements are requirement derived from the system under development. In order to manage the processes and support the processes execution, appropriate tools are useful, which assist developers with requirement and work product management. Work products are process outcomes representing different types of evidence. Evidence shows capability and maturity of the development process, compliance to the underlying standards and safety as well as security of the developed products. In addition, evidence is used to support arguments which are related to requirements.

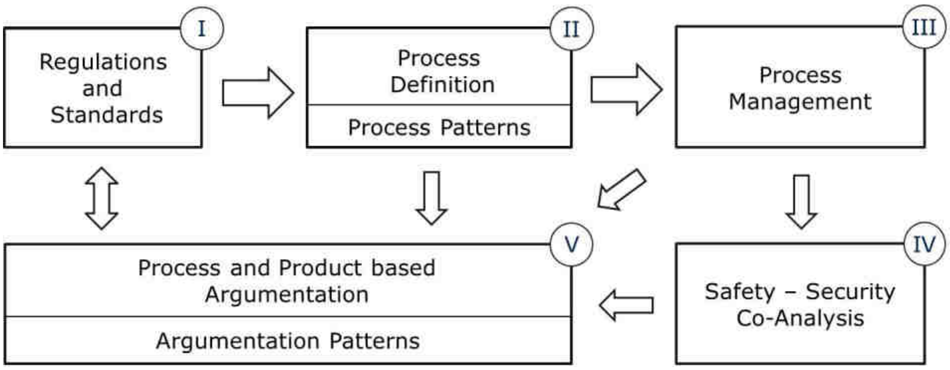

I
Regulations and Standards
II
Process Definition
Process Patterns
III
Process Management
IV
Safety – Security Co-Analysis
V
Process and Product based Argumentation
Argumentation Patterns

Fig. 2. Safety and security co-engineering framework

Safety and Security Co-Analysis (IV). The intention of the proposed framework is to integrate functional safety and security. For that reason we have to deal with special analysis methods (see section 3.2), which handle safety and security aspects in one common analysis (co-analysis) methodology.

Process- and Product-based Argumentation (V). Consequently the argumentation demonstrates that the item under consideration contains no unreasonable risk and consolidates functional safety and security. To visualize these relationships between requirements and work products we use GSN. A more detailed description of the argumentation approach can be found in [17], [18].

To recapitulate we consider a loop (depicted in Fig. 2) in which every activity is supported by a tool: We create processes which are modelled, instantiated and executed. The process output is evidence to argue that activities for the development of a specific product have been performed and are compliant to specific regulations. Once the process has integrated various disciplines, like safety and security, project managers have support to coordinate their cooperative actions.

3.1 Process Management

The requirements-driven workflow during process management starts with capturing requirements derived from the system artefacts, from standards, and possibly other, e.g. domain specific sources. The goal is a valid combined safety and security case, which requires evidences for the arguments it is composed of. The next step in the process is the definition of the necessary assurance activities, for which appropriate tools and methods are assigned. Finally, the assurance activities are processed - as far as possible automatically by a workflow engine. Successful assurance activities yield the necessary evidences. In case of negative results the faulty system element needs to be amended and then the assurance activity needs to be re-processed. When all assurance activities have been processed successfully the combined safety and security case is complete and valid.

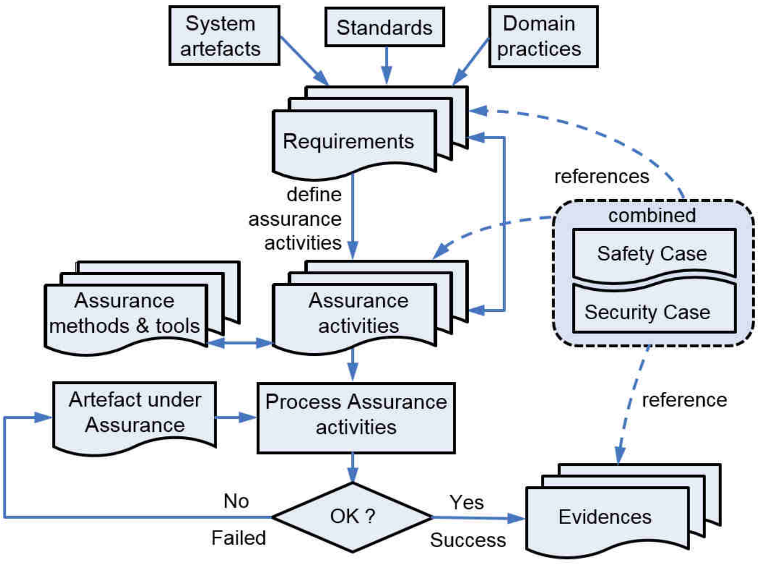


Fig. 3. Workflow model supporting compositional safety and security case

3.2 Safety and Security Co-Analysis

Integrated development processes have to deal with requirements concerning functional safety and security. They affect not only safety related methods (e.g. HARA), they also demand methods for joint safety and security analysis (e.g. STAMP, STPA-Sec, FMVEA).

STAMP approach is used in this framework for co-analysis to model systems as hierarchical structures. Higher level controllers in the hierarchy control the processes at lower levels via actors, while the lower levels send feedback to the higher levels via sensors. It provides support to identify root causes for accidents in modern complex systems. Therefore, safety accidents should be viewed as a result of a lack of control, instead of a chain or sequence of events (i.e. Swiss cheese model). System-theoretic Process Analysis (STPA) is a novel analysis approach derived from STAMP.

STPA uses a control theory based system consideration. STPA for Security (STPA-Sec) [19] extends the safety-focused method to cover security. In STPA-Sec, each control action is analyzed under different possible conditions and guidewords are used to identify loss scenarios. A loss is a situation of insufficient or missing controls or safety constraints. STPA-Sec consists of following steps:

- Step 1. System description (scope, control model, accidents and hazards).
- Step 2. Identification of unsafe control actions (using control actions from Step 1 and guidewords to identify unsafe control actions in all system states and environmental conditions). Control action not given, given incorrectly, wrong timing or order, stopped too soon or applied too long.
- Step3. Identification of scenarios which can cause unsafe control actions: identify scenarios how unsafe control action can be caused, based on control loop.
- Step 4. Design controls and countermeasures based on scenarios.

3.3 Patterns for Process and Argumentation

Patterns are a concept which spreads out in various development areas. We are using patterns to provide process and argumentation frameworks, which represents most of the recurring steps. The intention is to spend time once and reuse the elaborated patterns many times. Especially the integration of activities related to functional safety and security is a challenging work. We created patterns that provide process- and argumentation-templates. Process patterns simplify creating development processes because they already bring together functional safety and security activities. Argumentation patterns are corresponding to the process and exhibit the line of argumentation using the created work products. They include argumentation concerning functional safety and security and the interaction between them. Both types of patterns have to be instantiated for the specific development project. Instantiation for example means to select project specific methods like STPA-Sec for co-analysis. In parallel, the corresponding line of argumentation has to be selected. The purpose of creating patterns within the framework is to simplify the process definition, where the elaboration of evidence and adequate fitting arguments supports claims related to requirements.

4 Application to the Use Case

The automotive hybrid powertrain use-case provides the basis for the analysis of safety and security aspects based on state-of-the-art material[2]. An integral part of the hybrid powertrain system is the high voltage (HV) battery system, which consists of the battery management system (BMS), the battery satellite modules (grouping battery cells in modules and communicating via dedicated bus), and a fan control for cooling of the battery cells. The BMS is the main E/E system inside of an HV battery to power electric or hybrid electric vehicles. The BMS consists of several input sensors (see Fig. 5) for cell voltages, cell temperatures, output current, output voltage, and actuators like HV contactors for disconnection. This system is connected to various powertrain control units, the charging interface (enabling the communication with battery charging stations), the on-board diagnostic interface, and via a dedicated gateway to the vehicle infotainment systems (including the human machine interface and a wireless infotainment internet connection).

For the demonstration of the applicability of the presented co-engineering framework we had to use existing tools, which have been extended for specific needs of the presented approach:

EPF-C[3] (Eclipse Process Framework – Composer) is used for tool-support regarding the safety and security process modelling (II).

WEFACT (Workflow Engine for Analysis, Certification and Test) [16], web-based distributed platform for requirements-based testing with continuous impact assessment in order to support the safety case with evidences. Test workflow was extended to a workflow for safety certification and in the EMC² project the attribute of security was integrated (III).

XSTAMPP (eXtensible STAMP Platform) [20] is an Eclipse RCP (Rich Client Platform) based tool which guides users through the Safety and Security Co-analysis by STPA-Sec process and supports the modelling of control loops and the definition of constraints (VI).

OpenCert is an open source tool for product and process assurance/certification management to support the compliance assessment and certification of safety-critical systems in sectors such as aerospace, railway and automotive [15]. OpenCert supports creation of GSN structures and mapping of evidence to requirements demanded by underlying standards (V).

In the following, the main parts the framework in scope of the EMC² project will be described in more detail.

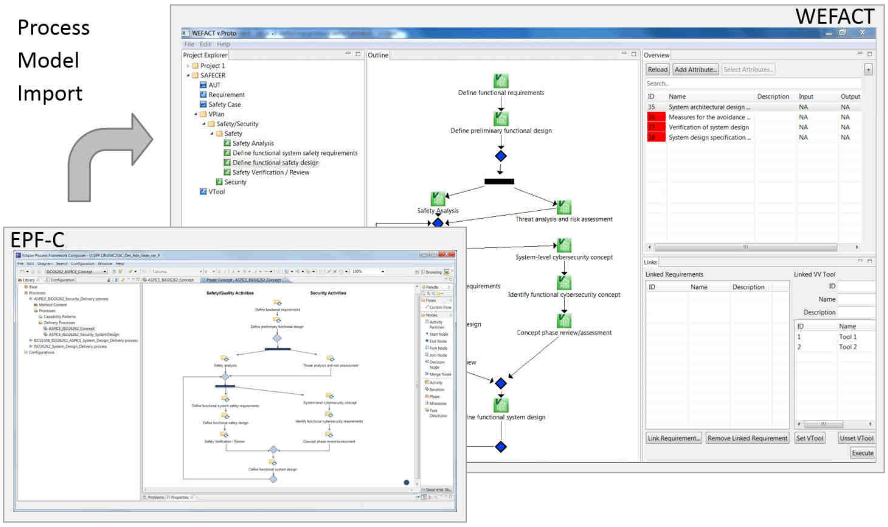
4.1 Process Definition and Process Execution

Efficient safety certification implies a process model which guides the user through the certification process and allows efficient compositional re-certification in the

[2] Technology-specific details have been abstracted for commercial sensitivity and presented analysis results are not intended to be exhaustive.

[3] Eclipse Process Framework, www.eclipse.org/epf/.

event of changes in the system. EPF-C provides elements to model phases and individual activities of the safety and security process. It allows modelling specific standards in a formal way, which enables automating the certification workflow.


Process
Model
Import
WEFACT
EPF-C
WEFACT <Proto
File Edit Help
Project Explorer
Project 1
SAFECER
AUT
Requirement
Safety Case
VPlan
Safety/Security
Safety
Safety Analysis
Define functional system safety requirements
Define functional safety design
Safety Verification / Review
Security
VTool
Outline
Define functional requirements
Define preliminary functional design
Safety Analysis
Threat analysis and risk assessment
System-level cybersecurity concept
Identify functional cybersecurity concept
Concept phase review/assessment
Define functional system design
Overview
Reload
Add Attribute...
Select Attributes...
Search...
Links
Linked Requirements
Linked VV Tool
Link Requirement...
Remove Linked Requirement
Set VTool
Unset VTool
Execute
Safety/Quality Activities
Security Activities

ID	Name	Description	Input	Output
35	System architectural design ...		NA	NA
36	Measures for the avoidance ...		NA	NA
37	Verification of system design		NA	NA
38	System design specification ...		NA	NA

ID	Name	Description

ID	Name
1	Tool 1
2	Tool 2

Fig. 4. Screenshots showing process modelling and execution (Tools: EPF-C, WEFACT)

WEFACT imports the process model including the activities modeled in EPF-C. **¡Error! No se encuentra el origen de la referencia.** shows safety- and security-related parts of the assurance process. The modeled assurance activities (small squares in the model diagram) are imported as so-called V-Plans and displayed as hierarchical list in the project explorer (left part of the GUI window). The upper right section of the window shows the assurance ("V&V") activities contained in the selected V-Plan. The V-Plan can be associated with the respective assurance tools (lower right corner). Finally, the assurance activities are processed by the workflow engine and deliver evidences for the requirements. During workflow execution, the status of the assurance activities changes whenever an activity is completed; the altered status is indicated by different highlighted colors in the list of activities.

4.2 Safety and Security Co-Analysis using STPA-Sec

The main accidents related to the BMS are fire/explosion of the battery systems and collision with an object:

- Fire / explosion of the battery system could be caused on the one side by charging conditions which are due to manipulation or failures outside of the safe range, but also by a modification or error in the operating parameters (e.g. spoofing on CAN bus, malicious firmware updates).
- Modified or erroneous operating parameter of the battery system or the control module, which provides power to the engines, could lead to undesired acceleration or deceleration. This could cause a collision.

Fig. 5 shows the representation of the system architecture in the XSTAMPP tool used for co-analysis. We focused on the control action "Charging Request" and identified the following unsafe control action, based on the guide phrase "Control Action given incorrectly": Excessive charging request is transmitted to charging unit during plug-in charging.

Potential safety and security scenarios for such an unsafe control action include:

- **Tampering**: An excessive charging request can be caused by a modified charging request from the BMS to the charging unit due to tampered process model in the BMS software to enable fast charging for not fast chargeable batteries. Potential motivation for the owner to hack his own car is that he is interested in faster charging and does not care about longevity of the battery due to leasing contracts.
- **Wrong Hardware:** A wrong charging request from BMS to charging unit may be caused by a failure/design error in the temperature sensor of the battery. Due to financial reasons, a manufacturer could reduce the number of sensors per battery module below the number required for a reliable reading. One additional scenario is that a maintenance provider uses sensors with lower resolution and hacks the control system to accept such sensors, which may be not certified for the task.
- **Manipulation**: Even when the vehicle BMS requests the correct power level a malicious manipulation on the communication between BMS and charging unit could lead to an unsafe charging request. Such a manipulation could be directed at the charging unit or at the BMS.

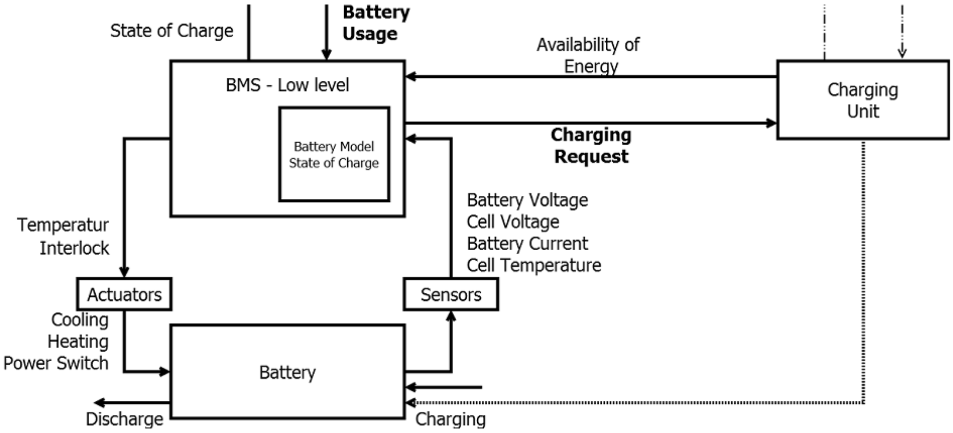


Fig. 5. Part of control loop of the battery management system (Tool: XSTAMPP) [21]

Supported by the tool XSTAMPP, we identify potential safety-related accidents based on potential causes regarding safety and security, e.g. failures and malicious manipulations by an attacker. In an independent analysis the focus of security would be on the classical CIA properties (confidentiality, integrity and availability). The feedback of safety relevance of these certain properties is missing. Safety specific analysis focuses only on safety issues caused by faults of E/E systems. Scenarios in which a user modifies the vehicle and causes a potential safety hazard would be missed. Co-Analysis connects the domains and supports the identification of safety goals and safety-related security goals.

4.3 Process- and Product- based Argumentation

Application of the methodology during the development of a BMS starts with selection of underlying standards. In this use case we consider ISO 26262 and SAE J3061 which are modeled in EPF-Composer as standard compliant integrated process model. The intention is to consider interacting functional safety and security activities. Based on the process model we examine the concept phase which includes the Hazard Analysis and Risk Assessment (HARA). Results of the HARA are “Automotive Safety Integrity Levels” (ASIL), safety goals to mitigate potential safety-critical hazards and high level safety requirements. The necessary process steps based on SAE J3061 have to be added to the existing process model. In other words, the process model based on ISO 26262 has to be extended with steps demanded by SAE J3061 to define a co-engineering process. Executing this process means to perform co-analysis using STPA-Sec method. One result of the co-analysis is the hazard "overcharging battery during plug-in charging" for which developers have to implement an adequate countermeasure. Overcharging will be possible if an attacker modifies the BMU parameters. To document the relationship between requirements (represented as goals) and measures (declared in evidence documents) we use the OpenCert GSN editor. On the one hand the argumentation covers the safety and security process and on the other hand it deals with the product specific decision how to prevent "battery overcharging". From the security process point of view the top level claim is "define functional cybersecurity requirements to prevent unauthorized changes to BMU parameters". These requirements are listed in the corresponding project specific document "HV_Batt_SecReq" stored in the project repository. From the product point of view the BMU needs capabilities to detect and prevent unauthorized change of parameters. The documentation of these capabilities is evidence and usable as product-based argumentation.

4.4 Results of investigation

The presented co-engineering framework was demonstrated by application to a hybrid electric vehicle powertrain use case. The application of the methodology showed a possible way how functional safety and security should correspond. Interaction between safety and security was forced by additional activities. The co-analysis method STPA-Sec was used and supported by the tool XSTAMPP. Product specific safety and security measures were coordinated to prevent unwanted interaction.

The usage of patterns speeded up the process definition activities and supported creation of argumentation fragments by GSN, which connect processes and evidence with argumentation. The graphical depiction of links between these elements improves the stakeholder’s understanding and shows how the dependencies between safety and security are organized. The tool OpenCert provides the possibility to manage patterns and to create GSN structures. The execution of the assurance activities by the workflow engine WEFACT allowed widely automated generation of evidences for the combined safety and security case.

5 Conclusion

Today's interconnected world needs special care to consider safety and security aspects. Although there are approaches treating the interaction between safety and security adequately they are still immature. This paper presented a safety and security co-engineering framework. A comprehensive combined safety and security argumentation methodology for the automotive domain has been developed. Its application in the automotive domain within the standards constraints provides useful information and can be considered as the next step for a wide application in development lifecycles. The following important benefits of the presented methodology for argumentation apply to the automotive domain: Usage of patterns improves process definition; the GSN structures connect process- and product-related evidence with argumentation; the graphical depiction of links between these elements improves the stakeholder's understanding of relevant safety and security aspects. In the HEV powertrain use case we showed the benefit of combined analysis of safety and security issues and the preparation of an assurance case for safety and security. The question, what is a compelling argument regarding the coordination of functional safety and security measures has not been answered in a satisfactory manner and needs further investigation.

The idea of safety and security co-engineering is becoming an accepted approach and it is required to appear in a specific standard regarding safety and security co-engineering activities and shall be treated in a normative manner. Experience gained in EU projects like EMC² and AMASS will try to reach standardization committees and influence developments of future editions of standards with the goal of supporting assurance case establishment.

Acknowledgment

This work is supported by the projects EMC² and AMASS. Research leading to these results has received funding from the EU ARTEMIS Joint Undertaking under grant agreement n 621429 (project EMC²), project AMASS (H2020-ECSEL no 692474; Spain's MINECO ref. PCIN-2015-262) and from the COMET K2 - Competence Centres for Excellent Technologies Programme of the Austrian Federal Ministry for Transport, Innovation and Technology (bmvit), the Austrian Federal Ministry of Science, Research and Economy (bmwfw), the Austrian Research Promotion Agency (FFG), the Province of Styria and the Styrian Business Promotion Agency (SFG).

References

1. Greenberg, A. (2015). Hackers remotely kill a jeep on the highway—With me in it. Wired, 7, 21.
2. Chen Yan, Wenyuan Xu, Jianhao Liu. (2016) Can You Trust Autonomous Vehicles: Contactless Attacks against Sensors of Self-driving Vehicle, DEFCON 24 conference

3. Borchert J., Slusser S. (2014, November). “Automotive (R)evolution: Defining a Security Paradigm in the Age of the Connected Car” Infineon Report Web http://www.infineon.com/car-security
4. Glas, B., Gebauer, C., Hänger, J., Heyl, A., Klarmann, J., Kriso, S., ... & Wörz, P. (2014). Automotive Safety and Security Integration Challenges. In Automotive-Safety & Security.
5. International Organization for Standardization. “ISO 26262 - Road vehicles – Functional safety, Part 1–10.” ISO/TC 22/SC 32 - Electrical and electronic components and general system aspects, Nov. 15, 2011.
6. SAE: J3061 Cybersecurity Guidebook for Cyber-Physical Vehicle Systems (2016)
7. Leveson, N. (2004). A new accident model for engineering safer systems. Safety science, 42(4), 237-270.
8. Macher, G., Sporer, H., Berlach, R., Armengaud, E., & Kreiner, C. (2015, March). SAHARA: a security-aware hazard and risk analysis method. In Design, Automation & Test in Europe Conference & Exhibition (DATE), 2015 (pp. 621-624). IEEE.
9. Schmittner, C., Gruber, T., Puschner, P., & Schoitsch, E. (2014). Security application of failure mode and effect analysis (FMEA). In International Conference on Computer Safety, Reliability, and Security (pp. 310-325). Springer International Publishing.
10. Goal Structuring Notation Working Group, GSN Community Standard Version 1, Nov. 16, 2011, www.goalstructuringnotation.info
11. Ray, A., & Cleaveland, R. (2015). Security Assurance Cases for Medical Cyber-Physical Systems. IEEE Design & Test, 32(5), 56-65.
12. Menon, C., Hawkins, R., & McDermid, J. (2009). Interim standard of best practice on SW in the context of DS 00-56 Issue 4. SSEI, University of York, Stand. of Best Practice (1).
13. Preschern, C., Kajtazovic, N., & Kreiner, C. (2013, October). Security analysis of safety patterns. In Proceedings of the 20th Conference on Pattern Languages of Programs (p. 12). The Hillside Group.
14. Taguchi, K., Souma, D., & Nishihara, H. (2015, September). Safe & Sec Case Patterns. In International Conference on Computer Safety, Reliability, and Security (pp. 27-37). Springer International Publishing.
15. Ruiz, A., Larrucea, X., & Espinoza, H. (2015, September). A Tool Suite for Assurance Cases and Evidences: Avionics Experiences. In European Conference on Software Process Improvement (pp. 63-71). Springer International Publishing.
16. Kristen, E., & Althammer, E. (2015, September). FlexRay Robustness Testing Contributing to Automated Safety Certification. In International Conference on Computer Safety, Reliability, and Security (pp. 201-211). Springer International Publishing.
17. Macher, G., Armengaud, E., Kreiner, C., Brenner, E., Schmittner, C., Ma, Z.,… Krammer, M. (in press) Integration of Security in the Development Lifecycle of Dependable Automotive CPS. In Druml, N., Genser, A., Krieg, A., Menghin, M., & Hoeller, A. (Eds.), Handbook of Research on Solutions for Cyber-Physical Systems Ubiquity. IGI Global
18. Martin, H., Krammer, M., Bramberger, R., & Armengaud, E. Process-and Product-based Lines of Argument for Automotive Safety Cases., ACM/IEEE 7th International Conference on Cyber-Physical Systems. (2016)
19. Young, W., & Leveson, N. (2013). Systems thinking for safety and security. In Proceedings of the 29th Annual Computer Security Applications Conference (pp. 1-8). ACM.
20. Abdulkhaleq, A., & Wagner, S. (2015). XSTAMPP: an eXtensible STAMP platform as tool support for safety engineering.
21. Schmittner, C., Ma, Z., & Puschner, P. (2016, September). Limitation and Improvement of STPA-Sec for Safety and Security Co-analysis. In International Conference on Computer Safety, Reliability, and Security (pp. 195-209). Springer International Publishing.

Table of Content

Chapter 1 Functional Safety of Automated Driving Systems: Does ISO 26262 Meet the Challenges?

Helmut Martin[1], Kurt Tschabuschnig[2], Olof Bridal[3], Daniel Watzenig[4]

Abstract: Today's innovative Automated Driving Systems (ADS) functions are realised by highly interconnected and networking cyber-physical systems based on existing Automated Driving Assistance Systems (ADAS). These interconnections increase the complexity of so-called systems-of-systems, because automation requires information and interaction with its environment. All possible interactions must be known for the definition of the intended system behaviour in order to identify any malfunctions of ADS, which may propagate over the system boundaries and influence other systems to fail in a harmful way. Hidden links are able to effect unwanted operational system states so that they can not be perceived as failure modes. For that reason, functional safety is an important topic for reduction of safety-critical risk to cause failures in complex automotive systems.

The chapter presented discusses the application of the automotive functional safety standard ISO 26262 in context of ADS. Following main topics are highlighted: Complexity of automated driving systems, issues concerning availability and reliability, importance of the concept phase and the role of the driver. Furthermore, proposals are made on how to handle these challenges and for feasible enhancements of the current ISO 26262 standard. Existing and promising methods are discussed that deal with the increasing complexity for the development of future ADS.

Keywords: ADAS, automated driving, functional safety, fail-safe, fail-operational, ISO 26262, safe state

1 Introduction

Science fiction stories about autonomous cars have inspired the imagination for many years. In early 1980s the television series *Knight Rider* presented the self-

[1] VIRTUAL VEHICLE Research Center, Graz/Austria

[2] MAGNA STEYR Engineering AG & Co KG, Graz/Austria

[3] VOLVO Group Trucks Technology, Gothenburg/Sweden

[4] Graz University of Technology, Institute of Electrical Measurement and Measurement Signal Processing, Graz/Austria

driving and artificial intelligent car named K.I.T.T.[5], and the slogan went, "Knight Rider – A shadowy flight into the dangerous world of a man who does not exist". Techies of the time were fascinated by the possibility of a technology and imagined that it would be possible to drive or simply travel in cars of the kind in the near future. Today, some decades later, that vision is starting to be made a reality, which will change and further influence the common understanding of the existing human road mobility system. For the last 30 years, the main innovations of vehicle technologies have been achieved by E/E systems in the automotive industry [1], e.g. Anti-lock Braking System (ABS) in 1978, Electronic Stability Program (ESP) in 1995 up to Collision Avoidance Systems in 2010 (see Figure 1).

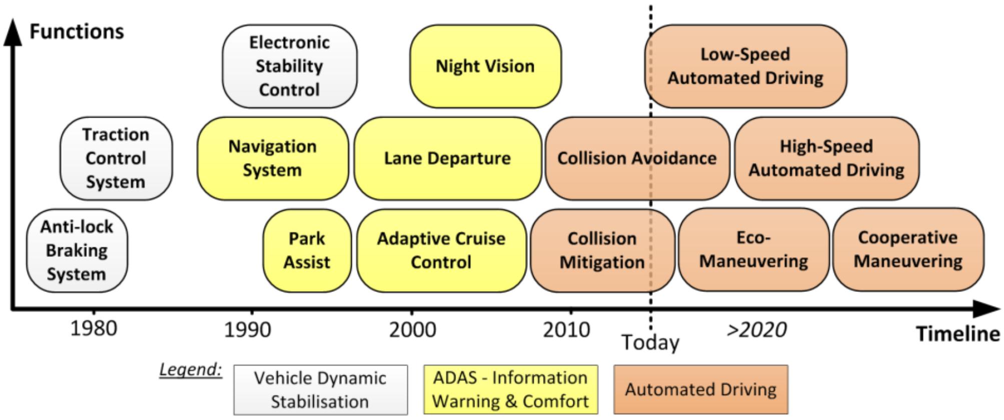


Figure 1: Evolution of advanced vehicle functions.

New generations of the Advanced Driving Assistance System (ADAS) are more complex than ever before in two aspects: firstly from a technical point of view in the context of the introduction of new technologies for implementing the functions required. Secondly from an organizational point of view concerning the whole supply chain including the suppliers involved for a different kinds of services and products during the lifecycle of an automotive vehicle. In this chapter, we will focus on the technical aspect as well as on the discussion about the challenges of automated driving functions and of how to apply the existing version of the ISO 26262 [5] standard concerning automotive functional safety.

1.1 From Driver Assistance to Highly Automated Driving Systems

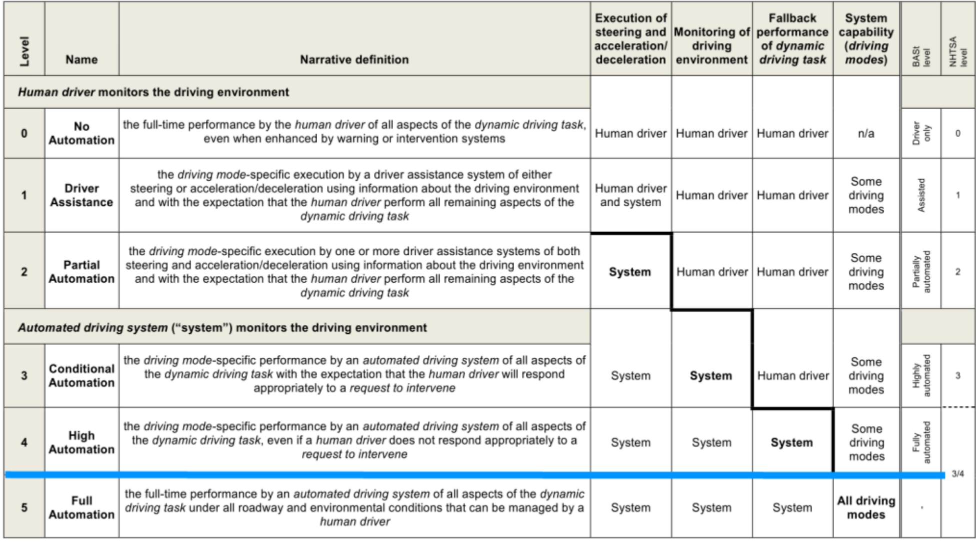
Today, almost every car on the market provides driver assistance systems (e.g. Electronic Stability Control – ESC). For safety reasons, high-class vehicles are equipped with various additional ADAS functions (e.g. Adaptive Cruise Control – ACC). The introduction of such systems has helped to reduce the number of fatal

[5] Knight Industries Two Thousand

accidents [7] [8]. However, more than 90 percent of accidents still occur as a result of human misbehaviour or mistakes. Thus, it is an important topic for the European Union to reduce the number of human-caused accidents by introducing the next generation of ADAS for our cars, which are referred to as Automated Driving Systems (ADS).

The different definitions of driving automation for on-road vehicles by SAE in the standard J3016 [3] and recommendations provided by BASt[6] and NHTSA[7] are shown and compared with each other in Table 1. The comparison between the levels proposed by the various standards/recommendations is possible up to the BASt Level 4 ‘fully automated’ (see blue line in Table 1).

Table 1: Definition of SAE Driving Automation levels for on-road vehicles and comparison with BASt and NHTSA [2].



Level	Name	Narrative definition	Execution of steering and acceleration/ deceleration	Monitoring of driving environment	Fallback performance of *dynamic driving task*	System capability (*driving modes*)	BASt level	NHTSA level
***Human driver* monitors the driving environment**								
0	**No Automation**	the full-time performance by the *human driver* of all aspects of the *dynamic driving task*, even when enhanced by warning or intervention systems	Human driver	Human driver	Human driver	n/a	Driver only	0
1	**Driver Assistance**	the *driving mode*-specific execution by a driver assistance system of either steering or acceleration/deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	Human driver and system	Human driver	Human driver	Some driving modes	Assisted	1
2	**Partial Automation**	the *driving mode*-specific execution by one or more driver assistance systems of both steering and acceleration/deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	**System**	Human driver	Human driver	Some driving modes	Partially automated	2
***Automated driving system* (“system”) monitors the driving environment**								
3	**Conditional Automation**	the *driving mode*-specific performance by an *automated driving system* of all aspects of the *dynamic driving task* with the expectation that the *human driver* will respond appropriately to a *request to intervene*	System	**System**	Human driver	Some driving modes	Highly automated	3
4	**High Automation**	the *driving mode*-specific performance by an *automated driving system* of all aspects of the *dynamic driving task*, even if a *human driver* does not respond appropriately to a *request to intervene*	System	System	**System**	Some driving modes	Fully automated	3/4
5	**Full Automation**	the full-time performance by an *automated driving system* of all aspects of the *dynamic driving task* under all roadway and environmental conditions that can be managed by a *human driver*	System	System	System	**All driving modes**	-	

Evolution of driving systems (based on the definition by BASt / Lx…Level x):

L0. *Driver only* - Driver assistance comfort system (e.g. speed limiter)
Responsibility: Driver
Safe State: Driver always has control of the vehicle

L1. *Assisted* - Advanced driver assistance provides safety improvement, ADAS supports the driver (e.g. EBA[8], ACC, LKA[9])
Responsibility: Driver
Safe State: Driver takes over full control of the vehicle

L2. *Partly automated* - Driving system controls laterally and longitudinally for a certain time in few situations (e.g. motorway assistant)

[6] Germany Federal Highway Research Institute (BASt) – http://www.bast.de

[7] US National Highway Traffic Safety Administration (NHTSA) – http://www.nhtsa.gov/

[8] Emergency Brake Assist

[9] Lane Keeping Assist

Responsibility: Driver
Safe State: Driver takes over full control of the vehicle

L3. *Highly automated* - Driving system controls lateral and longitudinal movement for a certain time in specific situations (e.g. motorway chauffeur)
Responsibility: Driver *OR* System
Safe State: Driver takes over full control of the vehicle within a specific timeframe *OR* System has to control the vehicle in defined driving situations, if the driver did not take over full control

L4. *Fully automated* – Driving system has complete control of lateral and longitudinal movement within a specified situation of the application (e.g. motorway pilot)
Responsibility: System
Safe State: System controls the vehicle in some driving situations

In SAE J3016, the highest level is 'Full Automation', which means from our perspective an autonomous vehicle that is able to drive without a driver. This level is not reached in this chapter because this scenario is too far away from today's technical practice.

The role of the driver will continue to be important for the introduction of automation functions in vehicles over the next few years. For high levels of automation the driver should not be required to cope with any critical driving situation. In such cases, the ADS should be able to handle any kind of driving situation autonomously – but this is still a future perspective expected that is expected to become reality around the years 2025–2035.

In the past, vehicle manufacturers realised their particular ADAS functions independently on a do it alone basis and using different OEM[10]-specific trade names (e.g. Adaptive Cruise Control (ACC), Active Cruise Control (ACC), Cooperative Adaptive Cruise (CACC), Distronic Plus). The function itself as well as the handling and the user interaction typically slightly differed from each other to guarantee OEM-specific originality. The levels of automation have to be harmonized for the introduction of ADS functions, otherwise the driver will not be able to operate different systems in the required way without training or a special extended driving license for automated vehicles as recommended by NHTSA [12]. One important aspect for handling the challenges is the standardization and harmonization of ADS functions of all OEMs on the market. The standardization must include not only the vehicle itself but also the overall aspects concerning the eco-system that are required to realise ADS functions like infrastructure (e.g. map data) or environment (e.g. secure C2X[11] communication). In aviation, the rulemaking advisory committee ARAC[12] harmonizes all the aviation-specific standards (e.g. for sys-

[10] Original Equipment Manufacturer

[11] Car-to-x means a communication between the car and any other external system, e.g. other cars C2C or the infrastructure C2I

[12] Aviation Rulemaking Advisory Committee – http://avstop.com/legal/2.htm

tem failures, underdetermined air traffic situation and human factor faults). The awareness of the need for such a rulemaking advisory committee for road vehicles is also given in the automotive industry as an automated vehicle will not be a closed system as was the situation in the past.

If we compare the situation of aviation with the road mobility standards concerning safety, ISO 26262 today covers only a subset of those system safety regulations. As an example, we wish to mention the interaction of ADS with the driver in aspects such as warning of the driver, supporting the driver so that an appropriate reaction can occur and feedback to the driver concerning his/her reaction. Only if the reaction of the vehicle is clearly defined and the driver knows which actions are carried out by the vehicle on its own, the right decision or reaction can be expected from the driver within a specific driving situation when needed.

1.2 Functional Safety according to ISO 26262

Safety is one of the key issues of road vehicle development. New innovative vehicle functionalities are not only introduced as driver assistance functions. Concerning propulsion, vehicle dynamics control as well as active and passive safety systems increasingly enter the domain of system safety engineering. Development and integration of these functionalities will enforce the need for a serious consideration of safety within the system development and the need to provide evidence that all reasonable system safety objectives are reached [3].

There are different Levels of Safety (LoS) (see Figure 2):

LoS1. *Safety with respect to product liability*[13] where safety aspects of any kind must be covered in order to achieve the permission for the launch of a product on a specific customer market (e.g. electrical safety of high voltage systems),

LoS2. *Functional Safety* with a cross-divisional view of any type of malfunction in mechatronic systems (e.g. failure of a mechanical part that could lead to an hazardous event),

LoS3. *Functional Safety* with emphasis on any kind of malfunction of electrical and/or electronic (E/E) systems (e.g. failure within the hardware which must be monitored and handled to achieve the safe state of a system). This means for the automotive industry, the ISO 26262 standard has to be applied.

[13] e.g. Austrian Federal Act - Governing the Liability for Defective Product/Product Liability [4]: §5. (1) A product §5. (1) A product shall be deemed defective if it does not provide the safety which, taking all circumstances into account, may be reasonably expected, in particular with respect to: 1.the presentation of the product; 2.the use to which it can reasonably be expected that the product would be put; 3.the time when the product was put into circulation.

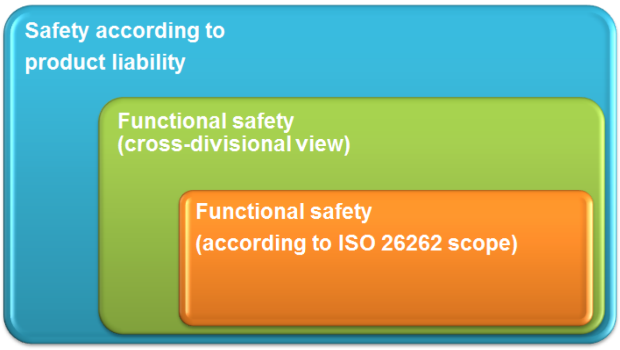

Safety according to product liability
Functional safety (cross-divisional view)
Functional safety (according to ISO 26262 scope)

Figure 2: Overview of different safety levels.

ISO 26262 "Road Vehicles – Functional Safety" is an automotive industry-specific derivation of the generic industrial functional safety standard IEC 61508 [6]. ISO 26262 was released in November 2011 as the state of the art international standard for E/E systems in passenger cars. It provides a structured and generic approach for the complete safety lifecycle of an automotive E/E system, including design, development, production, service processes and decommissioning. ISO 26262 defines the Automotive Safety Integrity Level (ASIL) as a risk classification parameter for the safety-critical hazardous situation of an item[14]. This is an important parameter for all subsequent safety activities in the safety lifecycle. The ASIL can be seen as a parameter that indicates a reduction of risk requirement in order to achieve a tolerable risk level.

The overall systems engineering must cover all kinds of system properties such as reliability, availability, maintainability, security and (functional) safety. Reliability engineering is closely related to safety engineering and to system safety. Both use common methods for their analyses and may require inputs from each other. Reliability engineering typically focuses on costs through failure caused by system downtime, cost of spares, repair equipment, personnel, and the cost of warranty claims. Safety engineering normally does not emphasize costs, but rather the preservation of life and nature. Therefore, it deals only with particular safety-critical and dangerous system failure modes [11]. Safety and reliability are different properties. A system can be reliable and unsafe while it can also be unsafe and reliable (see Figure 3). Furthermore, in some cases, these properties even come into conflict with each other. Leveson discusses this problem with very interesting examples from the military as well as the avionic and chemical industries [13].

[14] An item is a system or array of systems for implementing a function at vehicle level, to which ISO 26262 is applied.

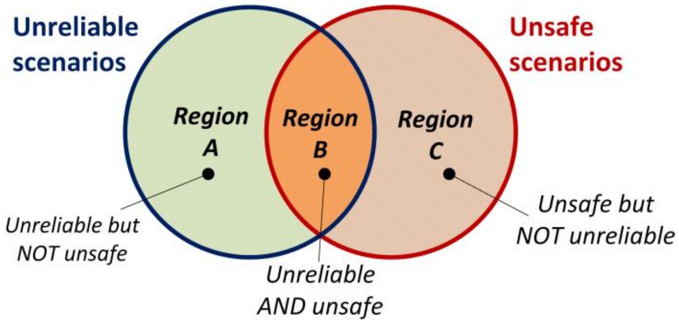


Figure 3: Relation of unreliable and unsafe scenarios.

The ISO 26262 standard states *"ISO 26262 does not address the nominal performance of E/E systems, even if dedicated functional performance standards exist for these systems (e.g. active and passive safety systems, brake systems, Adaptive Cruise Control)."* ASIL is not a nominal performance metric for other system properties (e.g. maintainability, reliability, availability) of ADS functions. Specific metrics for other concerns need to be examined in certain analyses of the particular scope (e.g. Mean Time To Repair (MTTR) for maintainable systems).

The ISO 26262 standard provides guidance by introducing requirements and recommendations to reduce the risk of systematic development failures and to handle the complexity of E/E systems. Nevertheless, compliance with the standard presents a significant challenge for companies, because ISO 26262 sets requirements and recommendations but does not explicitly define how they should be implemented in an efficient way in the context of a particular application. To implement the requirements and recommendations of the ISO 26262 in a particular application, expert knowledge in functional safety must create a thoughtfully argued and documented interpretation of the ISO 26262 for the particular application.

ISO 26262 provides a systematic top-down engineering approach based on the V-model[15]. A specification starts from the system-of-systems (SoS) level down to the sub-system and component level and subsequently to the implementation level of hardware (HW) and software (SW) modules. After the implementation and verification of HW and SW, the integration a bottom-up approach follows on at the right side of the V-model: integration of HW and SW modules in components (e.g. HW+SW in ECU), components in sub-systems (e.g. ECU in HV battery), sub-systems to system (e.g. HV battery in powertrain), system in SoS (e.g. powertrain in vehicle).

2 General Challenges of ADS

Some challenges are particularly relevant for automated systems in general terms (compared to 'classic' automotive electronic systems) and are related to complexity, availability and reliability. This section provides an overview of different kind

[15] See definition at http://v-modell.iabg.de/v-modell-xt-html-english/index.html

of challenges that must be investigated for the development of safety-critical aspects of ADS.

2.1 Increasing Complexity of ADS

A system can be described as an aggregation of elements or components concerning their cooperation and interaction with others to function properly. Interactions in a system are exchange processes between components realised by flow of material, energy and information (component relationships). In the event of failure, the system should be able to react in a fault-tolerant manner, which means that the system is able to trap a fault – "the system and its intended functions are able to survive" [9].

Safety is a system property intended to avoid system faults or malfunctions from causing any substantial damage (e.g. injuries to people or damage to the environment), which requires precise error detection. If an error is detected, the system must switch into a passive safe state with the consequence that the system is no longer available or reliable, but it is safe (failure integrity). The influence of system attributes such as availability, reliability, safety, security[16] must be harmonised and a kind of trade-off is required, because the ADS can be safe but that does not mean that the system is available or secure.

If a system is required to guarantee high availability and fail-operational characteristics, the system architecture is expected to have higher complexity of implemented functions. This means that the system grows in terms of the number of components and the interactions between them. The effort involved for the additional system safety causes increasing complexity. In addition unexpected effects arise when repetitive interactions are effected by increasing non-linear functions between the components. The most important attributes [10] of complex systems are:

- *Non-transparency* – state, interconnection and behaviour of a system and its components are only partly known,
- *Sensitivity* – interference of results in case of unexpected input changes,
- *Instability* – smallest disturbances cause unknown, unwanted behaviour of the system,
- *Internal dynamics* – continuous change of the system's state by the system itself without any external influence.

The mentioned attributes promote the appearance of additional faults and complicates their identification. Despite simplest components and interactions, the whole system generates forms, patterns and behaviour dynamics that could not be derived from particular components. This property is referred to as emergence[17], which arises from various signal feedbacks of the system components.

[16] See also "dependability" – umbrella term to describe different quality attributes of a system.

[17] Emergent entities (properties or substances) 'arise' out of more fundamental entities and yet are 'novel' or 'irreducible' with respect to them [36].

One popular development method is to abstract the reality, which means building a model to simplify or reduce the reality and capture the interesting major behaviour of the system. The state space of a model is always smaller than the state space of the real world because not all parameters such as temperature and friction that affect the components are considered. The synthesis of the component models does not show all operating states or all linking conditions. In particular, undesirable effects and hidden links could occur (see Figure 4).

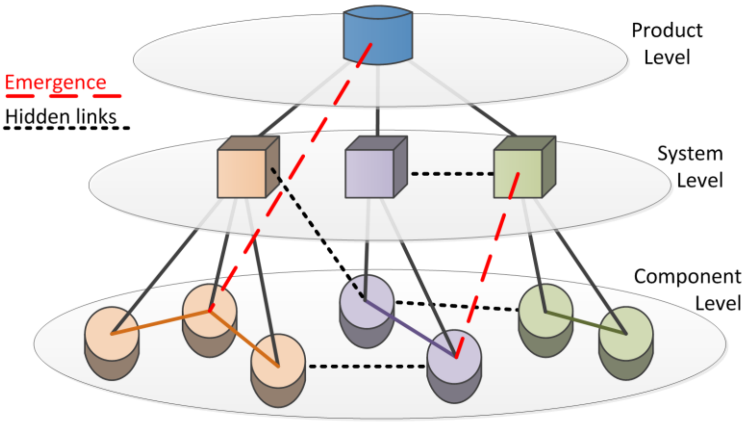


Figure 4: Latent linkages between system components and integration levels.

Unidentified coupling of components and over different integration levels may lead to systematic faults during the modelling of the systems. These non-transparency links and reactions to signals are the cause of unpleasant effects such as emergence (spontaneous system behaviour caused by smallest state changes on lowest level without direct derivation), common cause effects (single fault, cause simultaneously multiple components failure), powered run away (activation of a not provided function and are not designated in the conception or signal flow) and hidden links (unwanted operation states in the system, not identified as failure). In these cases, the system works incorrectly while a faulty state is not visible. That could cause the loss of all safety reserves in the system. The implemented safety mechanisms are ineffective and cannot be activated because the functional chain is unknown. These non-transparency links must be discovered during the system design.

The mission of mastering complex systems is to control the above-mentioned impacts in time and to prevent injurious effects. This could be done by a safe system design and increasing system transparency. The quality, robustness and fault tolerance of the design depends on prediction potential of the applied development procedure.

2.2 Strict Requirements concerning Availability and Reliability of ADS

A high degree of automation means that many - and potentially all - of the tasks usually carried out by the driver will now be executed by control systems in such a manner that the driver relies fully on the correct operation of these systems. The unavailability of a function – for example the inability to perform automated braking or automated steering – is more critical when the driver is 'not in the loop' than it would be if the driver is 'in the loop'. Regarding safety, it is generally considered as acceptable that a semi-autonomous function such as conventional cruise control or adaptive cruise control is suddenly deactivated, provided the driver is informed about the deactivation. The deactivation could be caused by a detected error in the system, by the activation of a stability function such as ESP or ASR or by some other triggering condition. The sudden loss of the vehicle's ability to drive autonomously, perhaps after several hours of fully autonomous driving, would typically be considered highly critical concerning safety, even if the driver is forced to take over control of the vehicle. In an extreme case, the vehicle continues to operate fully autonomously and to the extend that the driver does not even has any possibility to take over control.

Thus, the closer we approach towards fully autonomous vehicles, the more important it becomes to ensure that automated functions are fully available. Classical 'fail-safe' design solutions that rely on deactivating a function and informing the driver are no longer sufficient. Instead of fail-safe designs, fault-tolerant designs will be needed so that functions remain operational even when a fault is encountered in the system.

In context of criticality of potential failures of functions for highly automated driving, it is clear that systems providing the functions are able to significantly affect the vehicle behaviour. Potential failures can cause very bad effects and highly autonomous functions are, therefore typically associated with strict requirements on safety integrity. However, it should be noted that many conventional systems also require high levels of safety integrity, for example brake systems and steering systems. So, this aspect is not a *fundamental* difference between ADS functions and other vehicle functions. In general, automated functions tend to be associated with stricter safety requirements.

3 Challenges to ADS concerning Functional Safety

For relatively high levels of automation (i.e. closer to 'autonomous driving' than 'driver warning functions'), a complexity issue must be faced that makes the safety analysis more difficult than that of conventional systems. In a 'classic' vehicle, the driver is responsible for coordinating all the vehicle functions (propulsion, deceleration, steering, headlamps, direction indicators etc.). In principle, this means that each independent system function can be investigated separately with respect to functional safety and taking into account the possibilities that exist for the driver to handle a particular malfunction of that vehicle function. But with higher degrees of automation, the driver is no longer the overall coordinator, which means that any malfunction need to be handled by another function. In fact, the limits between these functions become blurred and difficult to define since the interaction between the different functions grows is now more complex. The ISO 26262 approach of looking at one function (or 'item', which is the real or imagined system that provides the function) at a time is less appropriate when the functions are heavily dependent on each other. In the following section, more safety-related topics will be discussed that must be taken into account for the engineering of ADS.

The innovations of today's vehicles follow a continuing evolutionary approach. The development of future technologies is based on existing automotive engineering best practices and does not only reuse the existing ones. Some of these evolutionary aspects will be discussed in the following.

3.1 Vehicle Platform for Basic Driving Functions

Many of the current discussions on ADS are concerned with the functional level to replace the single driver tasks by additional ADS functions. Further important issues that need to be covered are the basic actuation functions, such as accelerating, braking and steering, to implement the required vehicle movement. For these functions, today's vehicles provide function-specific assistance for the human driver through means such as force support in braking systems by a hydraulic or an electro-mechanic brake. Systems for automated driving functions need to be improved to support the fully required brake force without a human driver. Furthermore, the safety concepts of existing systems must be updated because the ECU (e.g. of the steering system) needs to detect any kind of malfunction and their effects have to be mitigated, because without a driver the system has to monitor, decide and react on its own. The steering system's safety goal can be formulated like, "Avoid the reversible and irreversible steering request from the steering system affected by any of the involved E/E systems" (e.g. steering angle sensor or ECU) [30]. The 3-Level Monitoring Concept (EGAS concept) provides a possible technical solution, which is a standardized principle for safety designs for vehicle engine controls published by German OEMs [31].

Future vehicle architectures will introduce new safety concepts in the automotive industry (e.g. steer-by-wire systems will change safety concepts in contrast to the systems nowadays). In the event of any fault, a deactivation in a fail-silent mode as a safe state will not be possible (e.g. a fail-operational mode can be realised by redundant system architecture). As a conclusion, it is obvious that the implementation of ADS functions in existing vehicle platforms cannot be seen as only add-ons to existing functions. The overall safety concept of vehicles has to be updated for upcoming requirements concerning fault-tolerant and fault-operational behaviour of highly automated vehicles.

Issue: Are existing vehicle platforms ready for ADS?

3.2 From ADAS to ADS Functions

Today, ADAS functions are used as a basis for the realization of ADS functions. However, these ADAS functions concern specific aspects of specific automotive use regarding

- *Scenarios*: from simple to complex scenarios (e.g. from keeping a driving distance by ACC on the Motorway to City Chauffeur at traffic crossing)
- *Vehicle speed*: from low to high speed (e.g. from Park Steering Assist to High-speed Motorway Chauffeur**)**
- *Vehicle Safety Risk*: from ‘normal’ to ‘low’ risk (e.g. from Emergency Braking Assist to Automated Driving on the Motorway)

The challenge is the combination and interaction of these basic functions. All kinds of interactions between these basic functions need to be analysed and handled in such manner that no unintended interactions concerning timing and value could occur. Any kind of functional and technical interaction must be dealt with during the system design phase.

Issue: Is reusing of existing ADAS possible?

3.3 Share of Sensors and Actuators

Different vehicle functions share the same sensors and actuators and all functional and technical condition has to be met. Sensor signals and actuator command signals may not be faulty in case of feature interaction and synchronization. In many applications an adequate fusion of sensor data and a voter mechanism for actuator command signals are required. In particular, any kind of unwanted interactions has to be handled so that no hidden links could affect any malfunction behaviour.

Issue: Is the available technology sufficient and adequate for the required functions?

3.4 From many ECUs to Host ECUs

Today, more and more functions of vehicles are implemented on existing single-core ECUs. These existing technologies slowly reach their limits (e.g. clock frequency, heat dissipation, size of gates). The following challenges approach is a shift from single-core to multi-core ECUs, which means a shift from distributed functions with many ECUs to a few multi-core host ECUs. The latter offer many different functions, but this rather new technology also requires new safety features. For safety-critical applications according to ISO 26262, these multi-core ECUs with shared resources have to support specific safety measures in hardware (e.g. use of lockstep core or memory protection). Furthermore, safety measures have to be supported by the software and software engineering constrains. Real-time (e.g. loads of cores), functional (e.g. sequences) and safety (e.g. spatial redundancy) aspects have to be considered by the operating system and the application software. Many new algorithms from different vendors have to be integrated in these platforms, and coordination, configuration and documentation pose a fur-

ther challenge. All these aspects have to be compliant to ISO 26262 and require safety evidence for the assessment of those applications.
Issue: Is new technology ready for safety-critical applications?

4 Importance of the Concept Phase

The concept phase defined in ISO 26262 focus on the functional abstraction of a specific item by (1) definition of the item, (2) assessment of the potential risks of that item by performing the Hazard Analysis and Risk Assessment (HARA), (3) determination of the ASIL for each hazardous event, (4) definition of high-level functional safety requirements as Safety Goals and (5) derivation of a Functional Safety Concept (FSC), which covers all relevant safety measures to achieve functional safety for the defined item. In the following, each of these activities is described and relevant steps will be discussed in more detail.

4.1 Item Definition

This activity covers the definition of the item, the required functionalities, the intended behaviour, the interaction with other items/systems of the vehicle and the interaction with the external environment of the vehicle. ISO 26262 is intended as an automotive-specific functional safety standard and it should be usable for any kind of E/E system in a vehicle. This can be slightly different when considered beyond the scope of specific items. For example, if we compare a hybrid powertrain system component such as a high-voltage battery system with an automated driving system for a Motorway Assistant (MWA): The MWA contains much more complex and networked functionalities that must to be coordinated with external items (e.g. other vehicles) and environmental systems (e.g. traffic signs) and furthermore with vehicle internal functions related to fundamental vehicle platform functions.

4.2 Hazard Analysis and Risk Assessment (HARA)

In the concept phase, the functional abstraction allows to have an abstract view of the system. Functional safety concerns unintended behaviour of the item. Safety analyses should be carried out in that phase to identify potential hazards of the item (e.g. HAZOP[18] or Concept FMEA[19]) followed by risk assessment.

The following steps describe activities that need to be done during the HARA including some proposed further extensions concerning ADS functions; these are written in bold letters and described in more detail:

[18] HAZard and OPerability study.

[19] Failure Mode and Effects Analysis.

Step 1: Elaboration of hazardous events

- Step1.1: Driving scenarios by situation analysis
  - Driving situation (e.g. maneuver at crossroads)
  - Infrastructure (e.g. communication between car and environment)
  - Environmental condition (e.g. weather)
  - Operating mode of the vehicle (e.g. acceleration)
  - Traffic participants involved (e.g. pedestrian)
  - **Driver presence** (e.g. driver in the loop/or not)
- Step 1.2: Hazard identification (e.g. by HAZOP)
  - From malfunctions
  - To malfunction behaviour
  - To hazard
- Step 1.3: Derivation of hazardous events
  - Combine driving situation with hazards
  - Potential source of harm to specific group of traffic participants at risk

Step 2: Classification of hazardous events

- Step 2.1: Severity classification
- Step 2.2: Exposure classification
- Step 2.3: **Controllability classification**

Driver presence and **controllability classification.** Each hazardous event is classified by the risk parameters severity (S), probability of exposure (E) and controllability (C) during the HARA. Parameter C denotes the estimation of controllability of a hazardous event by the driver or other persons potentially at risk. Controllability classes are C0 to C3, where C0 meaning "controllable in general" and C3 meaning "difficult to control or uncontrollable." In the specific context of risk assessment for automated driving functions, the parameters depend on the role of the driver within a specific driving situation, which is why an ASIL should be determined for any potential hazardous event. For ADAS and partially automated functions, the driver must always be able to take over control of the vehicle within a defined reaction time. Concerning functionality, for highly or fully automated functions, it is not required that the driver monitors the driving situation. Thus, it might not be possible for the driver to consider any kind of controllability of the vehicle. This may lead to a classification of C3, which would result in ASIL C/D[20] worst case.

4.3 Determination of ASIL and Safety Goals

The next steps concern the rating of ASIL and the definition of safety goals:
Step 3: ASIL derived from risk parameters

[20] Depending on the classification as S and/or E.

- ASIL = f (S, E, C) based on ISO 26262, part 3, Table 4

Step 4: Elaboration of Safety Goals

- Formulation of Safety Goals
- **Definition of Safety Goal attributes** (e.g. Safe State)

Definition of Safety Goal attributes. A safety goal must be specified as a top-level safety requirement. We want to avoid any unreasonable risk of a possible hazardous event (e.g. "unwanted acceleration shall not occur"). Safety goals are not expressed in terms of technological solutions but in terms of functional objectives. If a safety goal can be attained by transitioning to, or by maintaining of one or more safe states, then the corresponding safe state(s) shall be specified. Further relevant parameters regarding a safety goal are safe state, Fault Tolerant Time Interval (FTTI)[21], Diagnostic Test Interval (DTI)[22], Fault Reaction Time (FRT)[23] and Safe Tolerance Time (STT)[24] to maintain safe state before a possible hazard may occur (see Figure 5).

$$FTTI= DTI + FRT + STT$$

The definition of these parameters is very important in case of FRT being required to have critical driving situations handled by the system or by the driver to maintain the defined safe state (e.g. ADS function level 2 defines safe state as "driver takes over control").

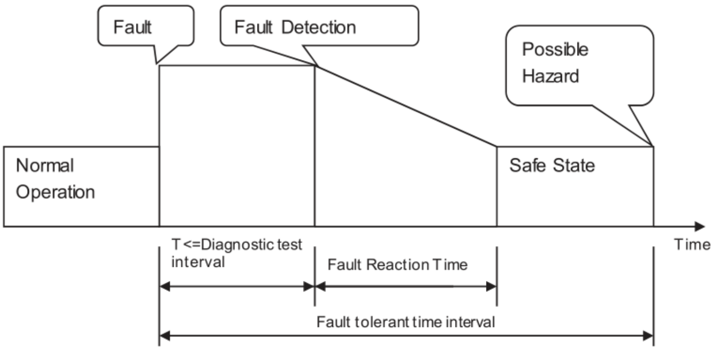


Figure 5: Fault Reaction Time and Fault Tolerant Time Interval [5].

[21] Time span in which fault(s) can occur in a system before a hazardous event ([5], Part3, 1.45).

[22] Amount of time in which a safety mechanism takes online diagnostic tests ([5], Part3, 1.26).

[23] Time span between detecting a fault and reaching the safe state ([5], Part3, 1.44).

[24] Amount of time between achieving the safe state before a hazard could occur.

Further influences to define a safe state. The complexity of the driving situation must be considered for the definition of safe states. Another important requirement in ISO 26262 concerning the safe state is **"8.4.2.4** *If a safe state cannot be reached by a transition within an acceptable time interval, an emergency operation shall be specified."*

Based on this requirement further constraints have to be taken into account:

- *Item Definition* – provided functionality of ADS to maintain safe state (e.g. low ADS level – only comfort functions vs. high ADS level – self-driving)
- *Driver Presence* – difference between driver in the loop or not (e.g. driver's hands on the steering wheel vs. checking e-mails at the touchscreen)
- *System Availability* – Possible or required degradation function depends on the level of ADS and the driver reaction in case of malfunction
- *Safe Place* – reachable safe place depends on the current driving situation and environmental conditions (e.g. safe state required during overtaking on the third lane of the motorway)
- *Safe State Scenario* – accessible safe state in specific driving situations including all constraints

An overview of different influences is given in Table 2.

Table 2: Overview of exemplary influences on the safe state.

Item Definition	LOW ADS	MID ADS		HIGH ADS
Driver presence	YES	YES	NO	NO
System Availability	Deactivation not available	Not available	Available	Available
Safe Place	–	–	Stop vehicle on the same lane	Stop at the rightmost lane
Safe State Scenario	Driver must take over	Driver must take over	Vehicle must stop at safe place	Vehicle must stop at safe place

4.4 Functional Safety Concept (FSC)

The objective of the functional safety concept is to derive functional safety requirements from the safety goals and to allocate them to preliminary architectural elements of the item or to external measures.

The following aspects have to be addressed in FSC:

- Error detection and failure mitigation
- Transition to a safe state
- Warning and degradation concept
- **Fault tolerance mechanisms**
- **Error detection and driver warning**
- **Arbitration logic**

The last three aspects will be discussed in the following in more detail:
Fault tolerance mechanisms means that a fault does not directly lead to the violation of the safety goal(s). The mechanism maintains the item in a safe state with or without any kind of degradation.
Error detection and driver warning is important to reduce the risk exposure time to an acceptable interval (e.g. engine malfunction indicator lamp, ABS fault warning lamp).
Arbitration logic is required to select the most appropriate control request from multiple requests generated simultaneously by different functions and is particularly important for the interacting functionalities of ADS.
However, not all of these aspects are always relevant for every system. Some systems do not offer any fault tolerance and some systems do not need any arbitration logic. The relevant safety measures concerning error detection, driver warning and transition to the safe state are important topics that must be considered in that phase.

4.4.1 Examples of FSC for Different ADS Levels

Depending on the type and degree of automation, there are several different strategies for ensuring safe operation despite faults in associated systems. This is illustrated in Figure 6, which shows three potential event sequences unfolding after the occurrence of an error. From top to bottom, these can be described as follows:

An **assisted or partially automated function** can no longer be trusted to fully function and as a consequence the driver is alerted to (re-)take control of the vehicle. During and after the handover, the partially automated function is prevented from working unsafely, perhaps by deactivating that function completely.

Example: Cruise control is deactivated due to a detected error. The driver is informed and takes control of the longitudinal motion of the vehicle.

A **highly or fully automated function** determines that the driver needs to take over due to a detected error. The driver is informed about the need for handover of control. Due to the expected relatively long time for the handover, the automated function needs to continue to operate fully or almost fully for some time.
Note: This means that the handover is initiated when the automated function is still either fully, or almost fully operational.
Example: An autonomous driving system detects an error that indicates that an <u>*additional*</u> *(subsequent) fault may lead to unsafe system behaviour. The driver is informed and takes control of the vehicle.*

A **fully automated function** without any possibility for the driver to take over control determines that the vehicle shall be stopped in a defined time interval to avoid any hazardous event. As in the previous case, this means that the handover is initiated when the automated function is still fully or almost fully operational.
Example: An autonomous driving system detects an error that indicates that an <u>*additional*</u> *(subsequent) fault may lead to unsafe system behaviour, so the auto-*

mated function brings the vehicle to a safe stop within a few minutes or possibly seconds.

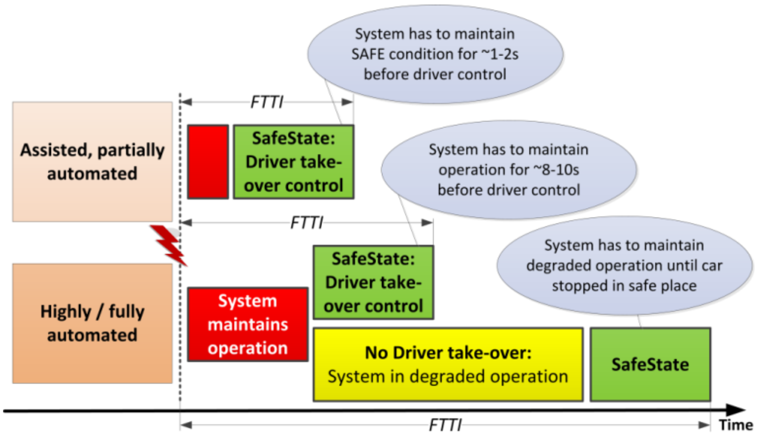

System has to maintain SAFE condition for ~1-2s before driver control
FTTI
Assisted, partially automated
SafeState: Driver take-over control
System has to maintain operation for ~8-10s before driver control
FTTI
SafeState: Driver take-over control
System has to maintain degraded operation until car stopped in safe place
Highly / fully automated
System maintains operation
No Driver take-over: System in degraded operation
SafeState
FTTI
Time

Figure 6: Different concepts for transition to safe state.

For the second and the third case described above, i.e. in the lower part of Figure 6, it is shown that the automated function needs to be fully - or almost fully - operational for several seconds after an error occurs. If there is no driver to take over, the function has to remain operational, albeit potentially degraded, for several minutes. Thus, the implementation of such highly or fully automated functions needs to be fault-tolerant in the sense that full or degraded functionality is possible even when a fault occurs in the system.

4.4.2 Vital Role of the Driver in the FSC

ISO 26262 sets requirements concerning error detection, driver warning and reaction of the driver. For today's automotive E/E systems, the role of the driver can be regarded as almost being covered in a cooperative manner. The driver must be able to control the vehicle on every trip (in Europe see also: Vienna Convention). By contrast, *how* the automated vehicles operate in a standardised way and *how* safety-critical aspects should be handled in a standardised way is *not* defined.

Thus, the driver needs to be familiar with different specific automated driving systems because the behaviour vehicles may differ. The training of the driver is required for specific ADS functions to ensure the driver's correct reaction within the required reaction time.

An additional aspect that must be taken into account here, and this is the 'habituation effect,' i.e. the introduction of ADAS and ADS functions will change the driving experience and require different skills of the driver. In HARA, the parameter C for controllability might change to 'uncontrollable'. In the near future, a driver may be unable to handle a critical vehicle situation without assistance systems within the required reaction time because of lack of experience. Special driving licenses for automated driving systems could be a possible scenario. However, they may not be accepted by customers who may hinder the introduction of such systems.

At present we do not train drivers to be able to deal with either a total brake failure or total loss of steering capabilities. Both braking and steering systems are extremely safe and reliable as a result so that the drivers do not need to worry about such problems at all. An alternative solution is simply to make the future ADS so safe and reliable that the drivers can fully rely on them at all times.

5 Supporting Methods to Handle Complexity of ADS

The complexity of these safety-critical systems must be considered and negative effects need to be detected and mitigated by fault identification and fault mitigation techniques. Today, in the development of automotive electronic systems there are established methods and technologies for safety activities available (e.g. Safety analyses such as HARA for ASIL determination [5], Failure Mode and Effect Analysis (FMEA) [32], Fault Tree Analysis (FTA) [33]).

The available technologies that need to be improved and developed further for their practical application in systems engineering:

- Formal/semiformal specifications by Model-Based Systems Engineering
- Formal verification by Contract-Based Design
- Simulation and Co-Simulation

5.1 Model-Based Systems Engineering (MBSE)

The following definition of MBSE can be found in Friedentahl [34]:
“Model-based systems engineering (MBSE) applies systems modelling as part of the systems engineering process … to support analysis, specification, design, and verification of the system being developed.”

The MBSE approach is a semiformal methodology to support engineers in the specification phase with analysis of the system and reduction of reality to an abstract model representation. The requirements for a specific level are defined and a virtual solution for the system is elaborated and hierarchically divided into representative components from system-of-systems, systems, sub-systems and components. Models at a lower hierarchy level provide more specific details concerning the realisation. During the modelling phase a separation of intended and unintended functions (= fault behaviour) is required, which is represented by specific functional properties and safety-related properties of the system. The model-based engineering approach is highly recommended by ISO 26262, part 6, for software development at ASIL C and D. This approach should be enhanced for the system level of such software-intensive systems. One of the major standardization working groups concering MBSE is the Object Management Group (OMG), which is an international, open membership, not-for-profit technology standards consortium. OMG Task Forces develop enterprise integration standards for a wide range of technologies and industries. Various standardised general purpose modelling languages are available for the system level (e.g. SysML[25], MARTE[26] or EAST ADL[27]). These modelling languages have been elaborated, improved, applied and evaluated by many EU research initiatives by academia, research and industry partners. MBSE presents many possibilities for how to model a system through the use of different modelling elements, but for practical application the reduction of the number of elements to a subset and provision of guidance and modelling constraints for engineers are requirements. A model-based systems engineering method[28] is a method that implements all or part of the systems engineering process and produces a system model as one of its primary artifacts. A system model provides the basis for specification of the intended behaviour of the system and is further used for identification and derivation of error models. An error model handles fault propagation over different hierarchy levels from singular components up to hazards at vehicle level. Different safety analysis methods (e.g. FTA or FMEA) can be supported by applying the error model. The output of the safety analysis defines safety measures by safety requirements for mitigation of any potential fault by detection, prevention, degradation or warning actions in the safety concept. A possible approach for the automotive domain by using SysML is described by Martin et al. in the SAE technical paper [16].

Biggs et al. [35] present a profile for a conceptional meta-model to cover relevant aspects of system safety and describes safety stereotypes based on SysML (e.g. Hazard, Harm, HarmContext,…). The profile models common safety concepts from safety standards and safety analysis techniques. As a profile of SysML, it can be used to directly model the safety-related information of a system in the same model as that system's design. Furthermore, the profile supports communication between safety engineers and system developers, in order to improve the understanding on both sides of the risks a system is vulnerable to and the features the system uses to mitigate those risks.

The MBSE approach by using SysML covers the following concerns [34]:

- *Provide a common and standardized description language* to improve the communication between system engineers and engineers from other disciplines.
- *Support of the performance of different kinds of checks* of the system model for the verification of specification rules (e.g. for the system design, to achieve correctness and completeness).

[25] Systems Modelling Language – http://www.omg.org/spec/SysML/.

[26] Modelling and Analysis of Real Time and Embedded systems - http://www.omgmarte.org/.

[27] Electronics Architecture and Software Technology – Architecture Description Language – http://www.east-adl.info/.

[28] A method is a set of related activities, techniques, conventions, representations, and artifacts that implement one or more processes and is generally supported by a set of tools.

- *Improve the processing of the system modelling artefacts* by using transformation of the system model to another description model and extension with other relevant aspects (e.g. error modelling).
- *Traceability of relevant safety artifacts* is provided and so the change management and impact analysis of particular safety concerns is possible. A further benefit of MBSE is the possible reuse of existing best practices by different kinds of patterns for requirements definition, safety design and safety argumentation.

5.2 Formal Verification by Contract-Based Design (CBD)

CBD is a formal method for specifying what a component/system is able to offer (e.g. service, data, information, energy) for its environment by means of so-called 'guarantees' and what a component/system requires (e.g. service, data, information, energy) from its environment by 'assumptions' [17]. Guarantees may be the performance and restrictions of output interface/channels which are only valid if all assumptions are confirmed. Assumptions defines the environmental constrains for the input interface or channel of a system or component. The coupling of software-intensive systems and their components is hard to handle. It is difficult to handle all potential hidden links that could affect the safety of a system. CBD is able to guarantee that the system model only engages defined system states. By applying CBD, only specified system states are allowed and the coupling and communication of systems is only permitted via defined and well-known channels.

It is possible to provide patterns to assume and guarantee contracts which are defined for different characteristic such as timing, safety, security etc., or patterns that are formalised to be checked automatically. The sum of all the system patterns defines all possible contracts.

CBD describes system components to be black boxes and defines their behaviour via interfaces with other system components. All kinds of dependability aspects are formulated as contracts; for example, timing (e.g. real-time contracts), safety (e.g. ASIL x or reaction time), security (e.g. authentication certificates) and are manageable by this means.

Different hierarchical levels of contracts are defined as follows, e.g.:

- Contracts between different SW modules
- Contracts between SW modules and HW components
- Contracts between different HW components
- Contracts between HW components and subsystems

CBD is able to coordinate interoperability and boundary limits of components and services they provide and also data over different hierarchical organisations. By modularization, it is possible to reduce the complexity of the components during system design. Every component is described by a limited catalogue of properties and constraints which establish safety. Conflicts between contracts are found very easily by means of a consistency test, if all contracts are free of any contradictions.

Satisfactory tests check whether the implementation of a component is consistent with the contract. Adequate tooling support is now finally available today (e.g. for model checking). Several publications discuss the use of contracts in context of the requirements of the engineering and safety standards such as ISO 26262 [18].

A new methodology to support the development process of safety–critical systems with contracts is presented by Baumgart et al. [19]. They compared existing meta–models also stating their short–comings in relation to their approach and they introduced the semantic foundation of our meta–model. They described their concepts of abstraction levels, perspectives, and viewpoints and provided a proof of concept with exemplary use cases.

Westman et al. [20] shows that safety requirements can be characterized by contracts for an item and its elements with guarantees that constitute the safety requirements, by providing explicit requirements on their environments as assumptions. A contract therefore enriches a safety specification for an item/element by explicitly declaring what each element/item expects from the environment to ensure that the safety requirements are satisfied. Furthermore, they showed that consistency and completeness of safety requirements can be ensured through verifying the dominance property of contracts.

Past and recent results as well as novel advances in the area of contracts theory are presented by Benveniste et al. [37]. They show that contracts offer support to certification by providing formal arguments that can assess and guarantee the quality of a design throughout all design phases. Furthermore, they showed that contracts can be used in any design process: Contracts provides an "orthogonal" support for all methodologies and can be used in any flow as a supporting technology in composing and refining designs.

5.3 Simulation and Co-Simulation

Simulation methods are commonly used in the automotive industry where complex embedded systems from different cooperative disciplines are referenced to realise highly interdependent functions. In this context, simulation methods allow engineers to predict the behaviour of complex embedded systems without an available prototype of the entire system. Complex systems like ADS require a data structure that considers the behavioural interactions within the system because of their multi-disciplinary nature. A combination of simulation and MBSE methodology supports modelling activities and improves the integration of simulation activities in the design process. This combination supports a system presentation for addressing the overall behavioural aspects of the product (multi-physics, local and global behaviours) and thus considers several system levels.

The ISO 26262 standard recommends the use of simulation methods for verification on different system integration levels (e.g. ISO 26262 part 3 for verification of the controllability parameter of HARA [25]). For system design verification, ISO 26262, part 4, Table 3 suggests simulation as a highly recommended method and a technique for e.g. fault injection and back-to-back test for ASIL C and D.

A model-based workflow for safety-critical embedded system is shown by Karner et al. [15]. Their approach covers three main aspects during the development of safety critical systems. Namely system modelling, system simulation and system verification based on simulation. By using the Software Process Engineering Metamodel (SPEM), the workflow is defined in a consistent and seamless way, allowing continuity from preliminary concepts up to the final system verification report. Aligned with requirements given by ISO 26262, the demonstrated workflow enables safety verification at system level during an early stage of development by using modelling and simulation.

A system modelling based approach for the integration and test of automotive embedded systems is proposed by Krammer et al. [14]. A V-model is introduced, targeting process oriented needs for safety and indicates where modelling languages in favour can be applied best. To establish a link between safety goals and the structure of simulation models, the initial model is enriched with necessary information and transformed to a language suitable for advanced simulation tasks. SystemC has the capabilities to support this approach for hardware and software even-handedly. The integration of SystemC into a co-simulation environment also enables the usage of external simulation models within the proposed architecture. The proposed system modelling based approach enables safety verification and validation at an early stage of development.

Graignic et al. [21] propose a software framework based on a data model that manages complex system structures. This data model structures behavioural information that considers three major interactions: interactions between components simulation models, interactions considering multi-level behaviours (e.g. use of components simulation for a module simulation) and interactions between domain behaviours (e.g. thermal impact on mechanical components) in a so-called co-simulation environment. Such methods can be used to perform early validation of the specifications by the MBSE approach to provide early validation feedback of adequate safety measures.

In the context of automated driving, different aspects beyond embedded systems behaviour are simulated such as the interaction of a vehicle with its environment, other vehicles or systems (e.g. Simulation of Urban MObility – SUMO [22] [23]) or the interaction of a vehicle with a driver, the interaction of vehicle subsystems for dynamic proof of a specified behaviour of systems and components [24].

6 Further Safety-Related Topics

In the following section, more safety-related topics will be discussed that must be taken into account for the engineering of ADS.

6.1 Influence of Security on Safety functions

One objective of system development is to ensure ‘freedom of unreasonable risks’ in any operational condition. This objective has different meanings depending on whether safety or security aspects are considered. From the safety point of view the risk to the environment arising from inside of the system must be minimized (and this apart from a system including humans). This can result in a technical failure in the system, for example fire hazard due to a high-voltage battery system of an electric vehicle or an accident because of an unintended acceleration of the ADS. Regarding security, potential threats to the system through the environment, which could result from intentional manipulations, e.g. a hacker attack, must be minimised. While the term safety represents the system view on any potential hazards of the system to the world outside, security concerns by contrast the aspects from the outside world to inside the vehicle and the influence on the vehicle internal systems. The goal of security measures is to protect the system from unauthorised use and manipulation (hacker, low-cost spare parts etc.). The discipline of security in the automotive industry concerns the growth in vehicle functions and the innovation potentials in the networking of vehicles with the environment (e.g. other cars) or Internet of Things (e.g. cloud services). The particular challenge on the one hand is the linking of the two disciplines safety and security for utilising synergies and on the other hand the prevention of conflicting effects. Different motivations for unauthorised access scenarios in vehicles are possible [29]:

- Manipulation of the vehicle and its components as well as the corruption or deactivation of vehicle functions – attacking of 'availability of a service' (e.g. change of torque limits of the electric machine that could damage the powertrain)
- Vehicle tuning by changing functional properties – attacking of 'functional integrity' (e.g. chip tuning, manipulation of the speedometer or deactivation of warning messages)
- Illegal attempts to obtain personal data – attacks on 'personal integrity' (e.g. the driving behaviour of the user, preferences for shops, restaurants or hotels)

ISO 26262 provides guidance for automotive development process issues concerning functional safety lifecycles. However, a process for security concerns is not state of practice for automotive engineering. Many similarities exist between safety and security on a common abstraction level and it would appear to be useful to interweave ISO 26262 development processes with security concerns. After defining a security item, the result of these considerations could be the consideration of security risks and the preparation of hazard analyses. Security goals with corresponding security measures can, hence, be derived from the analyses. After system design, verification and validation, a joint assessment should take place to rate the functional safety level reached according to ISO 26262 and any safety threat on the security side. Based on the similarities of these two disciplines, it would appear to be wise and necessary to expand the ISO 26262 framework by aspects of security topics. The extend to which these suggestions or other methods are expedient will be established in the course of an ongoing discussion in different standardisation communities [29].

6.2 Liability of ADS

Liability is a crucial topic in the context of future automated vehicles because legal authorities need an answer to the question, “Who was responsible?” in case of an accident.

Different responsibilities can be found under to law [26], e.g.:

- *Liability of the vehicle keeper*: Any operational risk in connection with an automobile is born by the vehicle keeper – ADS will not change the liability for the operation of automatic systems in motor vehicles.
- *Liability of the driver:* In damage event a fault of the driver is legally assumed (under civil law) until proof of the contrary is provided. In case of a fault of the ADS, the driver still has the option to insist on proof of exoneration.
- *Motor vehicle liability insurance:* If a harmed third party raises claims against vehicle keepers or drivers, they will be covered by the insurance – ADS will not cause any relevant change of the liability principles of the motor vehicle liability insurance.
- *Product liability of the manufacturer:* The OEM is liable if a defective product was brought to the market being subject to product liability. The OEM must provide evidence that the product was not defective and did not cause damage. The drivers must be instructed carefully in order to reasonably influence their expectations about the system’s capabilities and to encourage drivers to perform any necessary overriding functions. The safety of the system design is closely linked to the instructions given to the driver.
- *Liability of the infrastructure:* Future highly and fully automated vehicle functions will require precise data. These data will refer to local conditions too and will require a time stamp. The vehicle infrastructure should be able to provide all necessary information and is also liable for safe and secure functionality.

Ethical aspects will also play a role. In complex driving situations, events may occur that are difficult to handle by human drivers and that could lead to so-called ‘dilemma situations’. Sometimes, it is not possible to manage critical situations without harm any people. Thus, a decision has to be made to determine the minimum of harm. A decision between “plague or cholera?” is a difficult one for humans to make, but it is even more difficult for machines. Future highly and fully automated systems will need certain risk determination algorithms that can rise to situations of this kind.

For this reason an ‘event data recorder’ in the vehicles will be a requirement for recording relevant information about crashes or accidents. Information from these devices is collected and analysed after a crash to help in determining exactly what happened. This will be similar to the ‘black box’ found in airplanes, which records all critical data in the course of a flight. Further research is needed for the assessment and classification according to the level of abstraction and degree of automation for a standardized definition and understanding.

6.3 Validation of ADS Functions

Systematic testing methods are very important for the validation of ADS functions (e.g. concerning safety aspects). For such complex systems, test methods must comprise a combination of simulation and real-world testing for different levels of integration like xiL (x in the loop) and model/software/processor/hardware/vehicle in the loop approaches. The most widely used approaches for the validation of driving functions are based on the V-model, endurance testing, xiL testing, open-loop offline perceptions tests, 'Trojan horse' tests, stepped implementation tests, complex tests and so on. All these testing methods have different potentials and disadvantages, for example 'Trojan horse' tests are functional tests without hazardous effects in serial cars [27].

A further issue of ADS functions is that a strategy for safety confirmation cannot be implemented because a malfunction mechanism cannot be caused by the function but by decisions of the system. Although a test is able to characterize safety-relevant system states. There could be system reactions during automated driving situation where the decisions cannot be affected by the ego-vehicle alone. The actions and reactions of other road users must be anticipated, but a one hundred per cent expectation cannot be ensured. Adequacy here cannot yet be reached on basis of road user reaction models. The system reaction is going to be probabilistic and the decision on accuracy will become time-dependent and ascertainable only in simple situations. The first development of automated function was concentrated on technology goals. But without appropriate validation steps for safety-critical automated functions the vehicles cannot hope to be established on the consumer market.

7 Conclusion

The ISO 26262 standard is intended to be an automotive functional safety standard for handling hazards caused by malfunctioning behaviour of E/E safety related systems including interaction of these systems. It does not address the nominal performance of E/E systems such as powertrain control or any kind of ADAS. For this reason the ISO standard is also applicable to any level of automated driving. But the complexity of such systems is much higher than today's engineers are used to deal with, because of the high degree of networking functionalities that must be handled. Different kinds of challenges must be considered to realise ADS functions in an adequate manner. Following challenges were discussed in this chapter: Increasing complexity of highly interconnected functions and influence of system attributes, such as availability, reliability, safety, and security must be harmonised.

The concept phase of ISO 26262 becomes more important for ADS functions because the development of ADS requires the engineering approaches and technologies beyond state of the art. In particular, influence of the driver in the HARA, definition of safety goals and corresponding attributes for specific levels of ADS (e.g. safe state) as well as the changes of the functional safety concept from fail-safe to fail-operational strategies.

Today, several methods are available to support complex systems but they must be improved for the development of ADS. Possible technologies were discussed to handle the increasing complexity: Model-Based Systems Engineering, formal verification by contract based development, as well as simulation and co-simulation. Which of those methods are adequate and applicable to meet a specific safety-critical demand still has to be defined and argued in the individual safety cases with respect to the specific context.

An enhancement of ISO 26262 that provides guidance for handling such highly complex systems would be useful. In the near future, that kind of application-specific guidance has to be discussed within the working group of ISO 26262 for the up-coming enhancement of the standard. This enhancement should be included in the upcoming revision of the standard which is scheduled to be released by the beginning of 2018. In particular part 3 of the standard needs additional guidance to classify hazardous events during the Hazard Analysis and Risk Assessment to determine the ASIL and the system level activities to handle highly networked systems.

8 Acknowledgements

The research work has been funded by the European Commission within the EMC^2 under the ARTEMIS JU. The authors acknowledge the financial support of the COMET K2 -– Competence Centres for Excellent Technologies Programme of the Austrian Federal Ministry for Transport, Innovation and Technology (BMVIT), the Austrian Federal Ministry of Science, Research and Economy (BMWFW), the Austrian Research Promotion Agency (FFG), the Province of Styria and the Styrian Business Promotion Agency (SFG).

9 References

[1] K. Bengler, et al. “Three decades of driver assistance systems: Review and future perspectives.“ In *Intelligent Transportation Systems Magazine*, IEEE 6.4, 2014, pp. 6-22.

[2] B. Walker Smith. (2013, Dec. 18). *SAE Levels of Driving Automation*. The Center for Internet and Society at Stanford Law School [On-line] Available: http://cyberlaw.stanford.edu/loda [2015, Oct. 12]

[3] SAE International. “SAE J3016 – Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems.” J3016-201401, Jan. 01, 2014.

[4] Austrian Federal Act “Governing the Liability for a Defective Product (Product Liability Act).” Jan. 21st, 1988, [On-line] Available: www.ris.bka.gv.at/Dokumente/BgblPdf/1988_99_0/1988_99_0.pdf [2015, Oct. 12]

[5] International Organization for Standardization. “ISO 26262 - Road vehicles – Functional safety, Part 1–10.” ISO/TC 22/SC 32 - Electrical and electronic components and general system aspects, Nov. 15, 2011.

[6] International Electrotechnical Commission. “IEC 61508 - Functional safety of electrical/electronic/programmable electronic safety-related systems.” 2nd edition, TC 65/SC 65A - System aspects, April 04, 2010.

[7] European Commission (2015, March). *CARE Project: Road Safety Evolution in the EU* [On-line] Available: http://ec.europa.eu/transport/road_safety/pdf/observatory/historical_evol.pdf [2015, Oct. 12]

[8] O. Carstena, et al. “Vehicle-based studies of driving in the real world: The hard truth?” In *Accident Analysis and Prevention*, 58, 2013, pp. 162-174.

[9] H. Butz, “Safety and Fault Tolerance in a Complex Human Centred Automation Environment.” Innovation Forum Embedded Systems, Munich, April 24, 2009 [On-line] Available: http://bicc-net.de/events/innovation-forum-embedded-systems [2015, Oct. 12]

[10] H. Butz. ”Systemkomplexität methodisch erkennen und vermeiden.” In *Anforderungsmanagement in der Produktentwicklung*, Roland Jochem, Katja Landgraf (Hrsg.), Symposion Publishing GmbH, Düsseldorf, pp. 183-217, 2011.

[11] R.W.A. Barnard, (2008), “What is wrong with Reliability Engineering?” INCOSE International Symposium, 18: pp. 357–365. [doi: 10.1002 /j.2334-5837.2008.tb00811.x]

[12] National Highway Traffic Safety Administration (NHTSA) (May 30, 2013) ”Preliminary Statement of Policy Concerning Automated Vehicles”, [On-line] Available: http://www.nhtsa.gov/staticfiles/rulemaking/pdf/Automated_Vehicles_Policy.pdf [2015, Oct. 12]

[13] N. Leveson. (2012, January) “Engineering a safer world: Systems thinking applied to safety.” MIT Press, [On-line] Availble:https://mitpress.mit.edu/books/engineering-safer-world [2015, Oct. 12]

[14] M. Krammer, H. Martin, et al. “System Modeling for Integration and Test of Safety-Critical Automotive Embedded Systems.” No. 2013-01-0189, SAE Technical Paper, 2013.

[15] M. Karner, et al. “System Level Modeling, Simulation and Verification Workflow for Safety-Critical Automotive Embedded Systems.” No. 2014-01-0210, SAE Technical Paper, 2014.

[16] H. Martin et al. “Model-based Engineering Workflow for Automotive Safety Concepts.” No. 2015-01-0273, SAE Technical Paper, 2015.

[17] B. Meyer (1992). “Applying ‘Design by Contract’.” In *Computer, IEEE,* 25(10), pp. 40-51, 2015.

[18] J.-P. Blanquart, et al. “Towards Cross-Domains Model-Based Safety Process, Methods and Tools for Critical Embedded Systems: The CESAR Approach.” In *Computer Safety, Reliability, and Security,* Volume 6894 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, pp. 57–70, 2011.

[19] A. Baumgart, et al. “A Model-Based Design Methodology with Contracts to Enhance the Development Process of Safety-Critical Systems.” In *Software*

Technologies for Embedded and Ubiquitous Systems, Volume 6399 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, pp. 59–70, 2011.

[20] J. Westman, et al. “Structuring safety requirements in ISO 26262 using contract theory.” In *Computer Safety, Reliability, and Security*, Springer Berlin Heidelberg, pp. 166–177, 2013.

[21] P. Graignic, et al. “Complex System Simulation: Proposition of a MBSE Framework for Design-Analysis Integration.” In *Procedia Computer Science 16*, pp. 59-68, 2013.

[22] D. Krajzewicz. “Traffic Simulation with SUMO – Simulation of Urban Mobility.” In *Barceló,Jaume (Ed.): Fundamentals of Traffic Simulation*, Series: International Series in Operations Research & Management Science, Vol. 145, Springer Berlin Heidelberg, 2010.

[23] J. Erdmann (2014) “Lane-Changing Model in SUMO.” German Aerospace Center [On-Line] Available: http://elib.dlr.de/89233/1/SUMO_Lane_change_model_Template_SUMO2014.pdf, [2015, Oct. 12]

[24] A. Rousseau, et al. “Electric Drive Vehicle Development and Evaluation Using System Simulation.” In *Proceedings of the 19th IFAC World Congress*, pp. 7886-7891, 2014.

[25] M. Fischer, et al. “Modular and scalable driving simulator hardware and software for the development of future driver assistance and automation systems.” In *New Developments in Driving Simulation Design and Experiments*, pp. 223-229, 2014.

[26] T. M. Gasser “Legal consequences of an increase in vehicle automation.” Bundesanstalt für Straßenwesen, 2013 [On-Line] Available: http://bast.opus.hbznrw.de/volltexte/2013/723/pdf/Legal_consequences_of_an_increase_in_vehicle_automation.pdf [2015, Oct. 12]

[27] H. Winner, W. Wachenfeld “Absicherung automatischen Fahrens.“ In *6.FAS-Tagung München*, Nov. 29, 2013. [On-Line] Available: http://tubiblio.ulb.tu-darmstadt.de/63810/ [2015, Oct. 12]

[28] H. Winner, et al. “ Handbuch Fahrerassistenzsysteme“, 3. Auflage, ATZ/MTZ-Fachbuch, Springer Fachmedien, 2015

[29] M. Klauda, et al. “Automotive Safety und Security aus Sicht eines Zulieferers”, Oct. 04, 2013, [On-line] Available: http://subs.emis.de/LNI/ Proceedings/Proceedings210/13.pdf , [2015, Oct. 12]

[30] D. Campos, et al. "Egas–collaborative biomedical annotation as a service." In *Proceedings of the Fourth BioCreative Challenge Evaluation Workshop*, Vol. 1, pp. 254-259, 2013.

[31] IAV GmbH - Ingenieurgesellschaft Auto und Verkehr “Standardized E-Gas Monitoring Concept for Gasoline and Diesel Engine Control Units.” Version 6, September 22, 2015 [On-Line] Available: https://www.iav.com/en/ publications/technical-publications/etc-monitoring-concepts [2015, Oct. 12]

[32] International Electrotechnical Commission “IEC 60812 - Analysis techniques for system reliability - Procedure for failure mode and effects analysis (FMEA).“ TC 56 – Dependability, Jan. 26, 2006.

[33] International Electrotechnical Commission "IEC 61025 - Fault tree analysis (FTA)." TC 56 – Dependability, Dec. 13, 2006.

[34] Friedenthal S., Moore A., and Rick S. "A practical guide to SysML: the systems modeling language." 3rd edition , Morgan Kaufmann, Nov. 7, 2014.

[35] G. Biggs, et al. "A profile for modelling safety information with design information in SysML." In *Software and Systems Modeling*, Springer, 2014.

[36] Stanford Encyclopedia of Philosophy, "Emergent Properties. " Feb 28, 2012, [On-Line] Available: http://plato.stanford.edu/archives/spr2012/entries/ properties-emergent [2015, Oct. 12]

[37] A. Benveniste, et al. "Contracts for System Design." INRIA, Rapport de recherche RR-8147, Nov. 2012. [Online] Available: http://hal.inria.fr/hal-00757488 [2015, Oct. 12]

Bibliography

[1] T. Nolte, H. Hansson, and L. L. Bello, “Automotive communications-past, current and future,” in *10th IEEE Conference on Emerging Technologies and Factory Automation, 2005*, vol. 1, p. 8, IEEE, IEEE, 2005.

[2] J. Fitzgerald, P. G. Larsen, and M. Verhoef, “From embedded to cyber-physical systems: Challenges and future directions,” in *Collaborative Design for Embedded Systems*, pp. 293–303, Springer, 2014.

[3] I. O. for Standardization, “ISO 26262 - Road vehicles - Functional safety Part 1-10,” tech. rep., ISO/TC 22/SC 32 - Electrical and electronic components and general system aspects, 2011.

[4] M. Schmidt, M. Rau, D. E. Helmig, and D. B. Bauer, “Functional safety - dealing with independency,legal framework conditions and liability issues.” http://www.sgs-tuev-saar.com/en/functional-safety.html, Aug. 2011. [Online; access 19-June-2017].

[5] S. International Electrotechnical Commission, Geneva, “IEC 61508 - functional safety of electrical/electronic/ programmable electronic safety-related systems (2005),” *IEC Standard*, 2005.

[6] V. QMC, “Automotive SPICE - International standard used in automotive industry.” http://www.automotivespice.com/, 2013. [Online; access 19-June-2017].

[7] Object Management Group (OMG), “Systems Modeling Language (OMG SysML) v1.3.” http://www.omg.org/spec/SysML/1.3/, June 2012. [Online; access 19-June-2017].

[8] B. des Innern Referat Presse, “V-model (2013).” http://www.cio.bund.de/Web/DE/Architekturen-und-Standards/V-Modell-XT/vmodell-xt-node.html, 2013. [Online; access 19-June-2017].

[9] C. Miller and C. Valasek, “Remote exploitation of an unaltered passenger vehicle,” *Black Hat USA*, vol. 2015, p. 91, 2015.

[10] G. Macher, H. Martin, and et.al., *Research on Solutions for Cyber-Physical Systems Ubiquity*, ch. Integration of Security in the Development Lifecycle of Dependable Automotive CPS, pp. pp. 383–423. IGI Global, 2018.

[11] A. I. Association, “pSafeCer Project Consortium.” https://artemis-ia.eu/project/30-psafecer.html, 2011. [Online; access 19-June-2017].

[12] A. I. Association, “nSafeCer Project Consortium.” Internet: https://artemis-ia.eu/project/40-nsafecer.html, 2012. [Online; access 19-June-2017].

[13] A. I. Association, “VeTeSS Project Consortium.” https://artemis-ia.eu/project/43-vetess.html, 2012. [Online; access 19-June-2017].

[14] A. I. Association, “EMC2 Project Consortium.” https://artemis-ia.eu/project/56-emc2.html, 2014. [Online; access 19-June-2017].

[15] E. Joint Undertaking, “AMASS.” https://www.ecsel.eu/projects/amass, 2016. [Online; access 29-November-2018].

[16] R. Messnarz, H.-L. Ross, S. Habel, F. König, and et.al., “Integrated automotive spice and safety assessments,” *Software Process: Improvement and Practice*, vol. 14, no. 5, pp. 279–288, 2009.

[17] R. Messnarz, I. Sokic, S. Habel, and et.al., “Extending automotive spice to cover functional safety requirements and a safety architecture.,” in *EuroSPI*, pp. 298–307, Springer, 2011.

[18] S. S. S. Institute, “SS-7740 - Road vehicles - Functional safety process assessment model,” tech. rep., Svenska Standard, 2012.

[19] S. T. Acuña and X. Ferré, “Software process modelling.,” in *ISAS-SCI (1)*, pp. 237–242, 2001.

[20] K. Z. Zamli and P. A. Lee, “Taxonomy of process modeling languages,” in *Computer Systems and Applications, ACS/IEEE International Conference on. 2001*, pp. 435–437, IEEE, 2001.

[21] R. Bendraou, J.-M. Jezequel, and et.al., “A comparison of six uml-based languages for software process modeling,” *IEEE Transactions on Software Engineering*, vol. 36, no. 5, pp. 662–675, 2010.

[22] OMG, “Software & systems Process Engineering Meta-model (SPEM) v2.0 - Full Specification,” tech. rep., Object Management Group, 2008.

[23] I. Ruiz-Rube, J. M. Dodero, M. Palomo-Duarte, and et.al., “Uses and applications of spem process models. a systematic mapping study,” *Journal of Software Maintenance and Evolution: Research and Practice*, vol. 1, no. 32, pp. 999–1025, 2012.

[24] B. Gallina, I. Sljivo, and O. Jaradat, “Towards a safety-oriented process line for enabling reuse in safety critical systems development and certification,” in *Software Engineering Workshop (SEW), 2012 35th Annual IEEE*, pp. 148–157, IEEE, 2012.

[25] T. Martınez-Ruiz, F. Garcıa, M. Piattini, and et.al., “Modelling software process variability: an empirical study,” *IET software*, vol. 5, no. 2, pp. 172–187, 2011.

[26] T. Ternité, “Process lines: A product line approach designed for process model development,” in *Software Engineering and Advanced Applications, 2009. SEAA’09. 35th Euromicro Conference on*, pp. 173–180, IEEE, 2009.

[27] Y. Dajsuren, M. van den Brand, A. Serebrenik, and et.al., “Automotive ADLs: a study on enforcing consistency through multiple architectural levels,” in *Proceedings of the 8th international ACM SIGSOFT conference on Quality of Software Architectures*, pp. 71–80, ACM, 2012.

[28] S. Friedenthal, A. Moore, and R. Steiner, *A practical guide to SysML: the systems modeling language.* Morgan Kaufmann, 2014.

[29] C. A. Ericson *et al.*, *Hazard analysis techniques for system safety.* John Wiley & Sons, 2015.

[30] N. Leveson and S. Goetsch, “Safeware: System safety and computers,” *Medical Physics-New York-Institute of Physics*, vol. 23, no. 10, p. 1821, 1996.

[31] I. E. Commission *et al.*, *Analysis Techniques for System Reliability: Procedure for Failure Mode and Effects Analysis (FMEA).* International Electrotechnical Commission, 2006.

[32] P. David, V. Idasiak, and F. Kratz, “Improving reliability studies with SysML,” in *Reliability and Maintainability Symposium, 2009. RAMS 2009. Annual*, pp. 527–532, IEEE, 2009.

[33] K. Thramboulidis and S. Scholz, “Integrating the 3+ 1 sysml view model with safety engineering,” in *Emerging Technologies and Factory Automation (ETFA), 2010 IEEE Conference on*, pp. 1–8, IEEE, 2010.

[34] E. Andrianarison and J.-D. Piques, “SysML for embedded automotive systems: A practical approach,” in *Conference on Embedded Real Time Software and Systems. IEEE*, 2010.

[35] J. Piques and E. Andrianarison, “Sysml for embedded automotive systems: lessons learned,” *Interfaces*, vol. 3, p. 3b, 2011.

[36] R. Mader, E. Armengaud, A. Leitner, C. Kreiner, Q. Bourrouilh, G. Grießnig, C. Steger, and R. Weiß, “Computer-aided PHA, FTA and FMEA for automotive embedded systems,” *Computer Safety, Reliability, and Security*, pp. 113–127, 2011.

[37] R. Mader, G. Grießnig, A. Leitner, and et.al., “A computer-aided approach to preliminary hazard analysis for automotive embedded systems,” in *Engineering of Computer Based Systems (ECBS), 2011 18th IEEE International Conference and Workshops on*, pp. 169–178, IEEE, 2011.

[38] R. Mader, G. Grießnig, E. Armengaud, and et.al., “A bridge from system to software development for safety-critical automotive embedded systems,” in *Software Engineering and Advanced Applications (SEAA), 2012 38th EUROMICRO Conference on*, pp. 75–79, IEEE, 2012.

[39] R. Mader, E. Armengaud, G. Grießnig, and et.al., “OASIS: An automotive analysis and safety engineering instrument,” *Reliability Engineering & System Safety*, vol. 120, pp. 150–162, 2013.

[40] R. Weissnegger, M. Schuß, C. Kreiner, and et.al., “Seamless integrated simulation in design and verification flow for safety-critical systems,” in *International Conference on Computer Safety, Reliability, and Security*, pp. 359–370, Springer, 2016.

[41] P. Feth, T. Bauer, and T. Kuhn, “Virtual validation of cyber physical systems.,” in *Software Engineering & Management*, pp. 201–206, 2015.

[42] M. Baird, “IEEE standard 1666-2005 SystemC language reference manual,” *IEEE Standards Association, New Jersey, USA*, 2005.

[43] M. Barnasconi, “SystemC AMS extensions: Solving the need for speed,” *DAC Knowledge center*, 2010.

[44] M. Karner, C. Steger, E. Armengaud, and et.al., “A cross domain co-simulation platform for the efficient analysis of mechatronic systems,” tech. rep., SAE Technical Paper, 2010.

[45] M. Krammer, M. Karner, and A. Fuchs, “Semi-formal modeling of simulation-based V & V methods to enhance safety,” in *Proceedings of the Embedded World 2014 Exhibition and Conference*, 2014.

[46] T. Blochwitz, M. Otter, J. Akesson, and et.al., “Functional mockup interface 2.0: The standard for tool independent exchange of simulation models,” in *Proceedings of the 9th International MODELICA Conference*, no. 076 in 9th International MODELICA Conference, (Munich; Germany), pp. 173–184, Linköping University Electronic Press, Sept. 2012.

[47] M. Association, *Functional Mock-up Interface for Model Exchange and Co-Simulation, Version2.0*, 2014.

[48] F. Corbier, S. Loembe, and B. Clark, “FMI technology for validation of embedded electronic systems,” in *Embedded Real Time Software and Systems*, 2014.

[49] W. Chen, M. Huhn, and P. Fritzson, “A generic FMU interface for modelica,” in *Proceedings of the 4th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools; Zurich; Switzerland; September 5; 2011*, no. 056 in International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools, pp. 19–24, Linköping University Electronic Press, 2011.

[50] C. Noll, T. Blochwitz, T. Neidhold, and C. Kehrer, “Implementation of modelisar functional mock-up interfaces in simulationx,” in *Proceedings of the 8th International Modelica Conference; March 20th-22nd; Technical Univeristy; Dresden; Germany*, no. 63 in International Modelica Conference, pp. 339–343, Linköping University Electronic Press, 2011.

[51] B. Pussig, J. Denil, P. De Meulenaere, and H. Vangheluwe, “Generation of functional mock-up units for co-simulation from simulink®, using explicit computational semantics: work in progress paper,” in *Proceedings of the Symposium on Theory of Modeling & Simulation-DEVS Integrative*, p. 38, Society for Computer Simulation International, 2014.

[52] U. Pohlmann, W. Schäfer, H. Reddehase, and et.al., "Generating functional mockup units from software specifications," in *Proceedings of the 9th International MODELICA Conference; September 3-5; 2012; Munich; Germany*, no. 076 in International MODELICA Conference, (Munich; Germany), pp. 765–774, Linköping University Electronic Press, 2012.

[53] H. Neema, J. Gohl, Z. Lattmann, and et.al., "Model-based integration platform for fmi co-simulation and heterogeneous simulations of cyber-physical systems," in *Proceedings of the 10 th International Modelica Conference; March 10-12; 2014; Lund; Sweden*, no. 096 in International Modelica Conferenc, (Lund; Sweden), pp. 235–245, Linköping University Electronic Press, 2014.

[54] A. Elsheikh, M. U. Awais, E. Widl, and et.al., "Modelica-enabled rapid prototyping of cyber-physical energy systems via the functional mockup interface," in *Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES), 2013 Workshop on*, pp. 1–6, IEEE, 2013.

[55] A. a. o. H. Office for Nuclear Regulation, "The purpose, scope, and content of safety cases, ns-tast-gd-051 revision." http://www.onr.org.uk/operational/techasstguides/ns-tast-gd-051.pdf, July 2016. [Online; access 19-June-2017].

[56] T. Kelly, I. Bate, J. McDermid, and A. Burns, "Building a preliminary safety case: An example from aerospace," *ROLLS ROYCE PLC-REPORT-PNR*, 1998.

[57] D. Standard, "Standard 00-55 (part 2) issue 2 requirements for safety related software in defence equipment, part 2: Guidance," *Glasgow: Ministry of Defence*, 1997.

[58] G. S. N. W. Group, "GSN community standard version 1." Internet: www.goalstructuringnotation.info, Nov. 2011. [Online; access 19-June-2017].

[59] J. Birch, R. Rivett, I. Habli, and et.al., "Safety cases and their role in ISO 26262 functional safety assessment," in *International Conference on Computer Safety, Reliability, and Security*, pp. 154–165, Springer, 2013.

[60] B. Gallina, "A model-driven safety certification method for process compliance," in *Software Reliability Engineering Workshops (ISSREW), 2014 IEEE International Symposium on*, pp. 204–209, IEEE, 2014.

[61] S. Wagner, B. Schatz, S. Puchner, and et.al., "A case study on safety cases in the automotive domain: Modules, patterns, and models," in *Software Reliability Engineering (ISSRE), 2010 IEEE 21st International Symposium on*, pp. 269–278, IEEE, 2010.

[62] I. Habli, I. Ibarra, R. Rivett, and T. Kelly, "Model-based assurance for justifying automotive functional safety," tech. rep., SAE Technical Paper, 2010.

[63] E. O. for the Safety of Air Navigation, "Safety case development manual, edition 2.2," techreport DAP/SSH/091, Eurocontrol, Nov. 2006.

[64] P. Mohagheghi and R. Conradi, "An empirical investigation of software reuse benefits in a large telecom product," *ACM Trans. Softw. Eng. Methodol.*, vol. 17, pp. 13:1–13:31, June 2008.

[65] A. Orrego and G. Mundy, "Srae: An integrated framework for aiding in the verification and validation of legacy artifacts in nasa flight control systems," in *Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International*, vol. 1, pp. 413–422, IEEE, 2007.

[66] G. Le Lann, *The Ariane 5 Flight 501 Failure-A case study in system engineering for computing systems.* PhD thesis, INRIA, 1996.

[67] F. Belli, "Assuring dependability of software reuse: An industrial standard," in *International Conference on Software Technologies*, pp. 72–83, Springer, 2013.

[68] P. Clements and L. Northrop, *Software Product Lines - Practices and Patterns.* Addison-Wesley, 2002.

[69] F. vd Linden, K. Schmid, and E. Rommes, "Software product lines in action: The best industrial practice in product line engineering. secaucus," 2007.

[70] C. Alexander, S. Ishikawa, and et.al., "A Pattern Language: Towns, Buildings, Construction (Center for Environmental Structure)," *Oxford University Press*, 1977.

[71] J. Vlissides, R. Helm, R. Johnson, and E. Gamma, "Design patterns: Elements of reusable object-oriented software," *Reading: Addison-Wesley*, vol. 49, no. 120, p. 11, 1995.

[72] A. Armoush, *Design patterns for safety-critical embedded systems.* PhD thesis, RWTH Aachen University, 2010.

[73] C. Preschern, N. Kajtazovic, and C. Kreiner, "Building a safety architecture pattern system," in *Proceedings of the 18th European Conference on Pattern Languages of Program*, p. 17, ACM, 2015.

[74] B. P. Douglass, *Real-time design patterns: robust scalable architecture for real-time systems*, vol. 1. Addison-Wesley Professional, 2003.

[75] B. P. Douglass, *Design patterns for embedded systems in C: an embedded software engineering toolkit.* Elsevier, 2010.

[76] L. L. Pullum, *Software fault tolerance techniques and implementation.* Artech House, 2001.

[77] H. Martin, S. Baumgart, A. Leitner, and D. Watzenig, "Challenges for reuse in a safety-critical context: A state-of-practice study," tech. rep., SAE Technical Paper, 2014.

[78] B. Gallina, S. Kashiyarandi, H. Martin, and R. Bramberger, "Modeling a safety-and automotive-oriented process line to enable reuse and flexible process derivation," in *Computer Software and Applications Conference Workshops (COMPSACW), 2014 IEEE 38th International*, pp. 504–509, IEEE, 2014.

[79] E. Kristen and E. Althammer, “FlexRay robustness testing contributing to automated safety certification,” in *International Conference on Computer Safety, Reliability, and Security*, pp. 201–211, Springer, 2015.

[80] E. Schoitsch, E. Althammer, H. Eriksson, and et.al., “Validation and certification of safety-critical embedded systems–the decos test bench,” in *International Conference on Computer Safety, Reliability, and Security*, pp. 372–385, Springer, 2006.

[81] H. Martin, M. Krammer, B. Winkler, and C. Schwarzl, “Model-based engineering workflow for automotive safety concepts,” tech. rep., SAE Technical Paper, 2015.

[82] M. Krammer, H. Martin, Z. Radmilovic, and et.al., “Standard compliant co-simulation models for verification of automotive embedded systems,” in *Languages, Design Methods, and Tools for Electronic System Design*, pp. 29–47, Springer, 2016.

[83] H. Martin, M. Krammer, R. Bramberger, and E. Armengaud, “Process-and product-based lines of argument for automotive safety cases,” *EM2 Summit, Vienna*, 2016.

[84] A. Ruiz, X. Larrucea, and H. Espinoza, “A tool suite for assurance cases and evidences: Avionics experiences,” in *European Conference on Software Process Improvement*, pp. 63–71, Springer, 2015.

[85] A. Ruiz, A. Melzi, and T. Kelly, “Systematic application of iso 26262 on a seooc: Support by applying a systematic reuse approach,” in *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*, pp. 393–396, EDA Consortium, 2015.

[86] H. Martin, R. Bramberger, and et al., “Safety and security co-engineering and argumentation framework,” in *SASSUR Workshop*, 2017.

[87] A. Abdulkhaleq and S. Wagner, “XSTAMPP: An eXtensible STAMP platform as tool support for safety engineering,” *Institute of Software Technology*, 2015.

[88] H. Martin, A. Leitner, and B. Winkler, “Holistic safety considerations for automotive battery systems,” in *Automotive Battery Technology*, pp. 1–17, Springer, 2014.

[89] L. Lam, “A practical circuit-based model for state of health estimation of li-ion battery cells in electric vehicles,” *Master of Science Thesis, University of Technology Delft. doi*, vol. 10, 2011.

[90] C. Unterrieder, M. Huemer, and S. Marsili, “Systemc-ams-based design of a battery model for single and multi cell applications,” in *Ph. D. Research in Microelectronics and Electronics (PRIME), 2012 8th Conference on*, pp. 1–4, VDE, 2012.

[91] H. Martin, K. Tschabuschnig, O. Bridal, and D. Watzenig, “Functional safety of automated driving systems: Does iso 26262 meet the challenges?,” in *Automated Driving*, pp. 387–416, Springer, 2016.

[92] M. Krammer, H. Martin, M. Karner, D. Watzenig, and et.al., “System modeling for integration and test of safety-critical automotive embedded systems,” tech. rep., SAE Technical Paper, 2013.

[93] M. Krammer, P. Stirgwolt, and H. Martin, "From natural language to semi-formal notation requirements for automotive safety," tech. rep., SAE Technical Paper, 2015.

[94] T. Amorim, H. Martin, Z. Ma, C. Schmittner, D. Schneider, G. Macher, B. Winkler, M. Krammer, and C. Kreiner, "Systematic pattern approach for safety and security co-engineering in the automotive domain," in *International Conference on Computer Safety, Reliability, and Security*, pp. 329–342, Springer, 2017.

[95] E. Armengaud, Q. Bourrouilh, G. Griessnig, H. Martin, and P. Reichenpfader, "Using the CESAR safety framework for functional safety management in the context of ISO 26262," *Embedded Real Time Software and Systems*, 2012.

[96] H. Martin, B. Winkler, A. Leitner, and et.al., "Investigation of the influence of non-e/e safety measures for the asil determination," in *39th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA), 2013*, pp. 228–231, IEEE, 2013.

[97] R. Mader, H. Martin, R. Obendrauf, and et.al., "A framework for model-based safety requirements round-trip engineering," in *System Safety and Cyber-Security Conference 2015, 10th IET*, pp. 1–6, IET, 2015.

[98] H. Martin, Z. Ma, C. Schmittner, D. Schneider, B. Winkler, G. Macher, M. Krammer, T. Amorim, and C. Kreiner, "Combined automotive safety and security pattern engineering approach (in press)," *Reliability Engineering & System Safety*, 2018.