

Helmut Martin, Z. Ma, Christoph Schmittner, Bernhard Winkler, Martin Krammer, Daniel Schneider, Tiago Amorim, Georg Macher, Christian Kreiner

Combined automotive safety and security pattern engineering approach

Journal article | **Accepted manuscript (Postprint)**

This version is available at <https://doi.org/10.14279/depositonce-9868>



Martin, H., Ma, Z., Schmittner, C., Winkler, B., Krammer, M., Schneider, D., Amorim, T., Macher, G., & Kreiner, C. (2020). Combined automotive safety and security pattern engineering approach. *Reliability Engineering & System Safety*, 198, 106773. <https://doi.org/10.1016/j.ress.2019.106773>

Terms of Use

This work is licensed under a CC BY-NC-ND 4.0 License (Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International). For more information see <https://creativecommons.org/licenses/by-nc-nd/4.0/>.

WISSEN IM ZENTRUM
UNIVERSITÄTSBIBLIOTHEK

Technische
Universität
Berlin

Combined Automotive Safety and Security Pattern Engineering Approach

H. Martin¹, Z. Ma², Ch. Schmittner³, B. Winkler¹, M. Krammer¹, D. Schneider⁴, T. Amorim⁵, G. Macher⁶, Ch. Kreiner⁶

Graz and Vienna, Austria - Kaiserslautern and Berlin, Germany

Abstract

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 workflow 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.

Keywords: ISO 26262, SAE J3061, Engineering workflow, Safety pattern, Security pattern, Automotive

¹Virtual Vehicle Research Center

²AVL List GmbH

³Austrian Institute of Technology GmbH

⁴Fraunhofer IESE

⁵Technische Universität Berlin

⁶Graz University of Technology

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 is an 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 the development of regular automotive systems, because a safety standard has been available since 2011. However, there is a lack of knowledge on how to handle related security aspects, which strongly gain importance given the trend towards CPS. The required security knowledge is either just non-existent, or, maybe even more often, distributed over different organizational units in a company and thus not easily accessible. Furthermore, there is no automotive specific security standard available at the moment.

Given the tight interconnection and the mutual impact of safety and security aspects, 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.

The safety domain has collected valuable experience in using model-based approaches for safety analysis [1] and integration of safety engineering with systems engineering by using a shared model as common viewpoint [2]. We extend this towards security and show how model-based engineering can support system, safety and security engineering.

In this paper, we specifically focus on the proper and due consideration of

30 the security aspect within the safety engineering life cycle, 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 aim to close the security knowledge gap by harvesting its manifold
35 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.

40 This paper is based on the conference paper “Systematic Pattern Approach for Safety and Security Co-engineering in the Automotive Domain” from Amorim et. al [3] presented at the main track of SafeComp2017⁷ in Trento/Italy. The figures in the methodology part and in the battery use case are taken from that publication. For this journal paper we provide more details regarding the
45 pattern description and more insight regarding the co-engineering methodology. Furthermore, we extended the application of the methodology on the battery management system case study regarding support of pattern engineering by further use case scenarios and emphasizing Model-based Systems Engineering (MBSE).

50 2. Background and Related Work

This section provides an overview 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

55 ISO 26262 (“Road vehicles – Functional safety”) [4] is an automotive domain-specific safety standard. It provides a structured and generic approach for the

⁷<http://safecomp17.fbk.eu/>

complete safety life cycle 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 for E/E systems. An Automotive Safety Integrity Level (ASIL) is used as a central metric for determination of development efforts required for the entire safety life cycle. For instance, the usage of established design patterns is recommended for all ASIL levels for each sub-phase of software development, as described in Subsection 4.4.7 in Part 6 of ISO 26262. Concerning security, the first edition, released in 2011, does not consider it explicitly neither there is any support or guidance. The second edition ISO 26262 (released by end of 2018) contains only some guidance where an interaction between safety and security may occur from a safety perspective. Part 4 & 6 for system and software engineering point for the requirement and architecture consolidation to the need to coordinate safety and security requirements. In addition Part 2 for the safety management requires communication channels between functional safety and disciplines related to functional safety like cybersecurity. We present here an approach which uses model-based engineering and patterns to formalize the interaction between safety and security for the system design and show an approach to address the new requirements regarding interaction.

SAE J3061 [5] is a cybersecurity process framework for secure development life cycle tailored to the automotive domain by using a corresponding V-Model, as defined in ISO 26262. In SAE J3061, safety and security interaction points were defined to coordinate the two engineering processes. In 2017 development of ISO/SAE 21434 (“Road vehicles – Cybersecurity engineering”) [6] has been started as a joint project between ISO and SAE. The goal is to develop a cybersecurity engineering standard for road vehicles that includes requirements for cybersecurity process and a common language for communicating and managing cybersecurity risk among stakeholders based on ISO 26262 and SAE J3061. It is planned to include similar guidance for the interaction between safety and cybersecurity from a cybersecurity perspective.

2.2. Safety and Security Co-Analysis and Co-Engineering

Safety and security aspect must be considered in a holistic way due to tighter interactions between safety and security in automotive CPS. In this paper, safety & security co-analysis refers to methods and techniques that can be used to identify safety hazards and security threats in a joint approach. Co-analysis includes activities in the early stage of the development life cycle, e.g. in the requirements engineering as well as the design phase. Safety & security co-engineering refers to engineering activities that consider both safety and security and their interactions in the development life cycle like trade-off analysis or shared testing, verification and validation. Furthermore, co-engineering considers all phases of the life cycle, in which co-analysis is an integral part.

In the context of automotive domain, existing analysis method Hazard Analysis and Risk Management (HARA), which is standardized in ISO 26262 for safety, can be extended with Threat Analysis and Risk Assessment (TARA) method, as mentioned in SAE J3061 to identify cybersecurity risks [7]. Other proposals include Failure Mode and Vulnerability Effect Analysis (FMVEA) [8] and Security Aware Hazard Analysis and Risk Assessment (SAHARA) [9] 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 life cycle according to relevant standards such as ISO 26262 and SAE J3061 based on the V-Model [10]. There are different approaches, partially supported by projects, to address and define a framework for co-engineering. One major approach is based on work done in SESAMO⁸ on interaction points, which is currently refined in the AQUAS⁹ project. The generic concept of interaction points is similar to our approach, defining points in the system engineering life cycle where engineering teams from different domains need to interact in order to build a system [11]. A systematic pattern based interaction between safety and security can be considered as a interaction

⁸<http://sesamo-project.eu>

⁹<https://aquas-project.eu>

115 point or, in the term of iterative refinement, an interaction cycle. Besides interaction point based co-engineering approaches there are also other approaches, like for example the approach which is developed in the AMASS¹⁰ project, which aims at a complete replacement of established development processes by an co-engineering process.

120 2.3. Architectural Patterns

Patterns are used to collect and organize solutions for 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 [12]. 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. [13] (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 [14][15], with some being just a variation of simpler ones. Armoush introduced in his PhD thesis [14] 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 [16][17] for hardware patterns and on Pullum [18] 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 [15]. They can address software or hardware issues and they allow systems to remain fully functional or to bring them to a safe state. Describing

¹⁰<https://www.amass-ecsel.eu>

existing patterns, but the ones used in the presented case study, is out of the scope of this work.

145 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
150 provide additional information for security validation.

Security patterns have attracted the attention of both academic researchers and industry [19]. 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
155 with UML models, in a hierarchically layered architecture or in a search-able pattern library. Security patterns have been proposed for requirements engineering, software system design such as web services, and Service-Oriented Architectures [20]. Open Security Architecture 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
160 and graphical architecture designs in a consistent template. In recent years, security patterns have also been proposed for CPS [21].

2.4. Model-based Systems Engineering

Model-Based Systems Engineering (MBSE) is an efficient approach to specify, design and analyze complex systems. Estefan et. al [22] provide an overview
165 of MBSE methodologies and cover a collection of related processes, methods, and tools used to support the discipline of systems engineering and the role of the system modeling language SysML in that context. SysML is a semi-formal, general purpose modeling language. It is based on Unified Modeling Language
170 (UML), constructed for systems engineering applications and is standardized

by OMG¹¹ [23]. 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. 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 diagrams; the behavioral diagrams are use case, state machine, activity diagrams, and sequence diagrams. [24]. We use in our approach the block definition diagram for the modeling of the system and the patterns. Model-based approaches are increasingly used for safety, this starts from safety analysis based on system models to the integration of safety artifacts into the system modeling [25]. Benefits of such an approach are the consistency between models, since there is only one system model with different views. There is also an ongoing development to extend Model-Based Systems Engineering towards security [26].

3. Methodology

Although patterns address specific problems, the context in which a pattern is applied influences how it should be applied. Therefore, practitioners require systematic guidance when using patterns to tackle safety and security problems, i.e. more than just a catalog of patterns is needed. We thus propose a model-based safety and security pattern engineering workflow 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. In addition, in order to support the described process in an optimal way, we introduce some specific extensions to the description of both types of pattern. These extensions focus on the links between the two engineering domains and

¹¹<http://www.omg.org/>

are thus a means to bridge the gap and come closer to a real co-engineering approach to support analysis and argumentation.

200 3.1. Pattern Description

With respect to safety pattern, extensions have been proposed to support engineers in adapting their assurance case argumentations after a pattern has been applied. Argumentation fragments, specified in Goal Structuring Notation (GSN), could, for instance, be provided specifically for each pattern [27, 15] .

205 For the presented approach we build on these safety pattern with argumentation guidance and propose to use something similar for security. However, in order to better support the interaction between safety and security and to foster co-engineering, we further enrich the descriptions with pointers to respective safety or security weak points related to the pattern and suggest application of
210 corresponding analyses to investigate these issues and generate evidences for the argumentation. In general, patterns act as a central base of knowledge, which should be maintained based on the lessons learned during their application. On the one hand positive examples and scenarios can be referred and on the other hand negative examples, where the use of a pattern raised new and unwanted
215 issues, should be documented as well.

Each pattern contains a general pattern description, which covers pattern name, pattern type, context, problem, forces solution, graphical representation. A generic pattern template is shown in Figure A.9.

220 The safety pattern adopts the generic template, correspondingly describes the safety content and additionally augments it with respect to security aspects. Key constituents of the safety pattern are:

- GSN fragment of safety pattern [15]
- General security implications (e.g. typical security properties that might
225 interact with the pattern)
- Guidance wrt. security analysis (e.g. in a voter pattern, a promising attack vector might be going through the voter, thus analysis of the voter

is recommended)

- GSN fragment of security implications [27]: The GSN fragment reflects the bullets above in a generic way, so that the integration of these security considerations into the argumentation is simplified/guided to a certain extent.

We design the security pattern to follow the same approach as the safety pattern, they adopt the generic template, describe the security content as well as additional information regarding potential safety implications. Key constituents of the security pattern are:

- GSN fragment of security pattern
- General safety implications (e.g. typical safety properties that might interact with the pattern such as hard real time requirements and encryption)
- Guidance wrt. safety analysis (e.g. introducing additional HW components in a functional path additional random HW failure have to be taken into account)
- GSN fragment of safety implication: The GSN fragment of the pattern reflects the bullets above in a generic way, so that the integration of the security pattern (as well as auxiliary safety analyses / evidences) is simplified/guided to a certain extent.

Appendix A of this paper provides examples of a safety pattern (see Figure A.10) and security pattern (see Figure A.11).

3.2. Generic Combined Pattern-based Engineering Workflow

The Generic Combined Pattern-based Engineering Workflow is the approach proposed in this paper to guide engineers selecting and applying safety and security patterns to develop safe and secure systems/products. This workflow is meant to be used in unison (and tightly integrated) with the usual safety and security engineering approaches. It therefore does not substitute established approaches but rather enhances them with further tasks. The approach is suitable to be used with existing patterns. The workflow can take place at the different

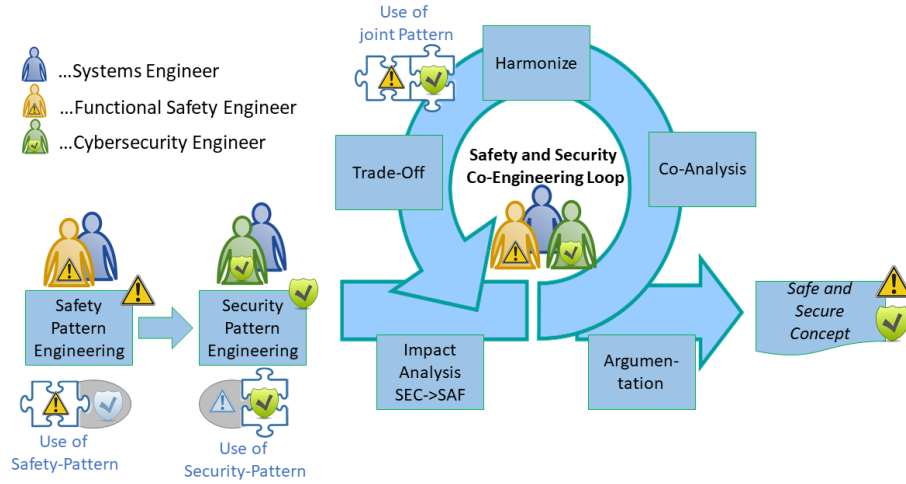


Figure 1: Generic combined pattern-based engineering workflow

development phases of the V-Model in framework of ISO 26262 [4] where design and specification are involved: concept phase, system level, hardware level and software level.

260 For example during the system design the Functional and Technical Concept are fully developed and both are used as input for the workflow.

The workflow is divided into three main phases happening one after the other in a waterfall fashion (cf. Figure 1): Safety Pattern Engineering, Security Pattern Engineering, and Safety and Security Co-Engineering Loop. In the first
265 two phases specific teams are involved: Safety Pattern Engineering is performed by systems and safety engineers, and Security Pattern Engineering is performed by systems and security engineers.

The output of the workflow is then consumed by the next phases of the V-Model, namely Product Development: Hardware level and Software level.

270

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 first safety architecture is almost defined.

In general, further changes in the architecture might open new vulnerability

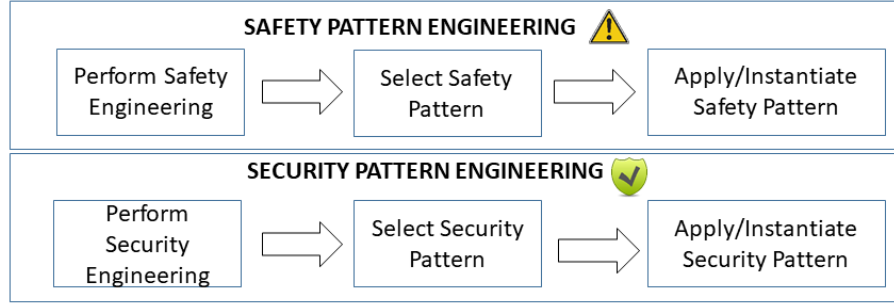


Figure 2: Safety pattern engineering and security pattern engineering tasks

points or might not be properly covered by mechanisms already implemented. However, it has to be taken into account that 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 workflow. In this phase all involved disciplines (systems, safety, and security engineers) work together in a joint approach. The loop prevents safety-motivated changes from creating unforeseen vulnerabilities and security-motivated changes from jeopardizing safety characteristics of the system. Each of these three phases will be described in the next subsection to provide a more detailed overview of the approach.

3.3. Safety Pattern Engineering

Safety Pattern Engineering involves safety-related tasks and is composed of three main tasks *Perform Safety Analysis*, *Select Safety Pattern*, *Apply/Instantiate Safety Pattern* (cf. Figure 2), which will be described in the following paragraphs.

Perform Safety Analysis. 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

295 available (i.e. Hazard and Risk analysis, FMEA and FTA to break down higher level to lower level requirements).

Select Safety Pattern. The decision about which pattern best fits a specific system should be analysed taking into account the problem to be addressed (typically ensuring identified Safety Requirements) and the context of the system. In
300 the proposed pattern template, particularly the information regarding pattern type, context, problem, forces and solution are now to be considered. Besides, there are a few trade-offs that one additionally needs to take into consideration when choosing an architectural pattern, such as costs (hardware, development effort) or standardization. Current state-of-the-art [14, 16, 17] provides many
305 patterns with detailed information about the impact in the system in the view of different dimensions (e.g. Cost, Reliability, Safety, Security), but some work is obviously required to create a comprehensive pattern catalogue that is fully in line with the templates proposed in this article. At any rate, there might always be cases that no pattern is suitable for the discovered problems and the
310 engineer needs to come up with an ad-hoc solution.

Apply/Instantiate Safety Pattern. Instantiating an architectural pattern for a system typically implies certain adjustments to tailor the pattern towards the existing system architecture and the problem to solve. In contrast, using the pattern as-is is usually not possible, because they have been developed out of
315 context and must thus be application and context agnostic. However, the patterns we propose (as described above and illustrated in the Appendix) provide comprehensive guidance regarding the activities to be performed as well as auxiliary material such as argumentation fragments to support the engineers in the best possible way. In addition, our safety pattern provide guidance with respect
320 to security implications, highlighting potential weak points and providing corresponding analysis support as well as argumentation support in form of a security related GSN fragment. In summary, the pattern instantiation does not only result in a revised system architecture (now including the elements modeled by the pattern) but also in an augmented safety and security argumentation bol-

325 stered by corresponding analyses. The updated system architecture as well as
the augmented analyses and argumentations are the prerequisites for the next
task.

3.4. *Security Pattern Engineering*

In the previous phase, the architecture was updated with safety measures,
330 the safety argumentation has been advanced and security implications of the
safety measures have been analysed. In the Security Pattern Engineering phase,
the architecture will be further analysed with regard to security vulnerabilities.
Corresponding pointers might be given directly by the descriptions (security
analysis and security-related GSN fragments) of the safety pattern selected in
335 the safety pattern engineering phase. Open weak points are to be addressed by
applicable security patterns. The output of this phase will be a secure architec-
ture.

Perform Security Analysis. In this step, Security Engineering is performed on
the existing system context such as functional requirements, results of Safety
340 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 ac-
345 ceptable level. Special attention is given to vulnerabilities introduced by safety
patterns.

Select Security Pattern. The security engineers should give priority to the se-
lection of re-usable security solutions from well-established security patterns for
mitigating the security risks. If multiple security patterns are available, the se-
350 lection 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 while taking
355 into consideration context of the system. The structure of the pattern and the instantiation approach is thereby similar to the safety pattern. In particular, beyond guidance regarding the application of the security pattern and corresponding security argumentation support, there are also pointers regarding potential safety implications which might play a role in the subsequent co-engineering
360 loop.

3.5. Safety and Security Co-Engineering Loop

After the initial two phases of the Pattern Engineering Workflow, the Safety and Security Co-Engineering Loop starts. The identified patterns cover safety and security measures defined by specific requirements. Each safety and security
365 measure should pass through the co-engineering loop to approve their adequate co-existence. Guidance with respect to the interrelationships is given by the pattern descriptions.

In this phase, lightweight versions of safety pattern engineering and security pattern engineering take place one after the other until no extra modification is
370 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 analyse how the newly added security patterns might impact the system safety. Some security
375 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
380 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
385 to be performed.

In the co-engineering loop iterative activities have been defined, which are described in the following and shown in a flow diagram in Figure 3 (Sx...Stepx, D...Decision):

390 *(S1.) Safety Impact Analysis.* After successful instantiation of the security pattern in the architecture, a safety impact analysis has to be performed. This analysis should check what happens if the introduced security measures may have any safety impact. But it is not only the safety failure of the security measure, it is also the possibility that the presence of that security measure
395 modifies the way the failures of any other elements of the architecture propagate in a modified manner, i.e. a complete safety analysis of failure modes of new/modified elements as well as redo safety analysis of other existing elements regarding any impact on existing failure modes.

(S1.D) Decision: Is there an impact of Security on Safety?

400 (S1.D.a) Answer NO - Safe and Secure Concept available

(S1.D.b) Answer: YES - Continue to with Co-Analysis

(S2.) Co-Analysis. Perform Co-Analysis for safety and security aspects by methods, such as SAHARA or FMVEA for identification of effected elements regarding new failure modes (safety) or new vulnerabilities (security).

405 *(S3.) Harmonization.* Joint engineering team elaborates and defines safety and security measures based on available patterns.

(S3.D) Decision: Is harmonization successful without any side-effect between safety and security measures?

(S3.D.a) Answer: YES - A possible solution has been elaborated. All safety
410 and security requirements are covered without any contradiction ("freedom of

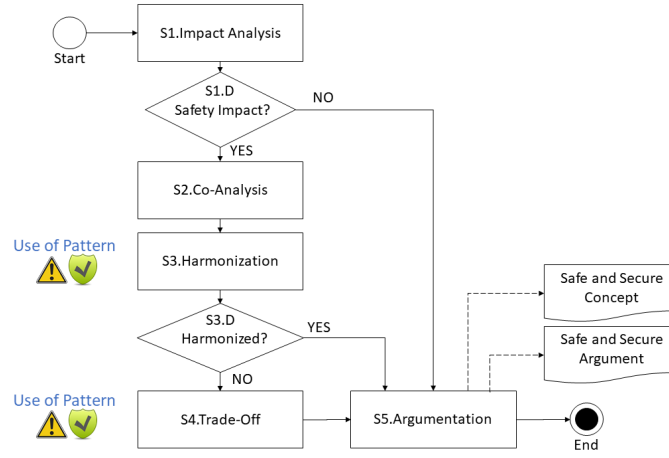


Figure 3: Flow chart of co-engineering loop iteration

interference”). Safe and Secure Concept is available. → Continue to (S5.) Argumentation.

(S3.D.b) Answer: NO - Proposals for different solutions are available. → Continue to (S4.) Trade-Off.

415 (S4.) *Trade-Off*. Joint engineering team has to prioritize which aspect is more relevant and a trade-off between safety and security measures have to be chosen. The trade-off can be supported by specific safety and security pattern, where a different pattern may be chosen to handle that issue and a compromise has to be achieved.

420 (S5.) *Argumentation*. The final activity of the loop will be Argumentation. In that task the safe and secure concept will be available and all rationals will be collected to compile the final safe and secure argument of the elaborated concept, that provides a piece to the Assurance Case.

4. Automotive Case Study

425 In the following section, the Combined Pattern-based Engineering Workflow is applied to an automotive case study, namely Electrified Hybrid Powertrain

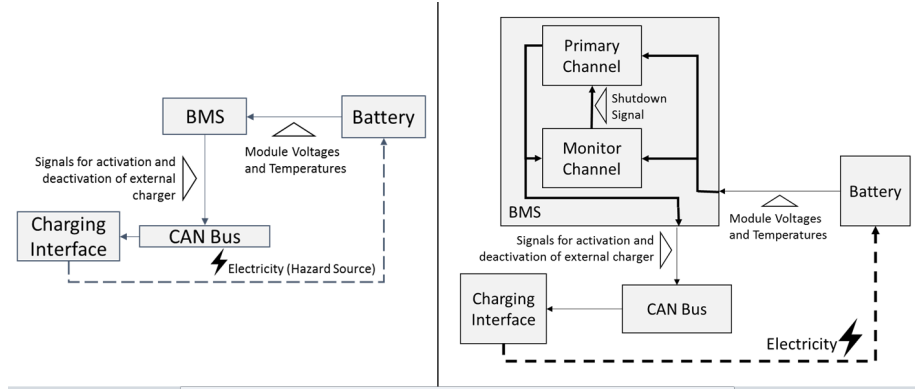


Figure 4: [Left]: Automotive Battery Use Case | [Right]: Architecture with the safety pattern applied

Battery Management System.

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 section, we investigate a specific use case scenario of the connected hybrid powertrain: charging of the battery system by connecting it with an external charging unit. Figure 4 [Left] shows the most relevant elements: battery (including all battery modules), BMS, CAN communication, charging interface, and external charging unit.

4.2. Application of the Approach

445 In this subsection, we apply the Combined Pattern-based Engineering Workflow presented in the previous subsection in the use case scenario in an early development phase called "concept phase".

The described approach to cover safety activities is based on the functional safety standard ISO 26262 and in particular the parameters (e.g. severity, exposure, controllability) used in HARA are taken from ISO 26262-part3, which 450 covers the concept phase.

For security ISO/SAE 21434 [6] is still in development and SAE J3061 was pushed back to work in progress. We use threat modeling as a well established security analysis method for the automotive domain [28], [29], [30]. In order to support a consistent engineering workflow we utilize a threat modeling add-in 455 for Enterprise Architect (EA) [31]¹². In this approach a model of the system is evaluated based on a formal description of threats to determine threats applicable to the analysed system. Different levels of the add-in are available. Furthermore, the MBSE tool EA provides good user support and possibilities 460 for a built-in pattern mechanism. EA defines patterns as a group of collaborating objects or classes that can be abstracted from a general set of modeling scenarios. Therefore, patterns are considered as an excellent means of achieving re-use and building in robustness. However, pattern modeling inherently introduces front-loading principles. A library of patterns needs to be built first, in 465 order to provide the engineer with a sufficient list of options to choose from.

4.2.1. Safety Pattern Engineering

Perform Safety Analysis. We describe in the following a small summary of the results of this task up to the level of Functional Safety Requirements:

470 **Hazard:** Wrong estimation of charging status.

Comment: The battery of electric vehicles can be very dangerous in case of

¹²www.sparxsystems.eu

overcharging, even causing explosions. If the charging status of a battery is estimated wrongly, extra energy might be supplied, leading to a hazardous situation.

475 **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.

480 **Hazard classification:** Severity: 3 — Exposure: 4 — Controllability: 2

Resulting hazard ASIL: [C]

Safety goal: Estimate correct status of cycle while charging.

Safe state: “HV Battery disconnected” AND “Driver alerted”.

Functional Safety requirement: Detect failures and errors from BMS.

485

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.

490 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 [16] (cf. Figure 4 [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.

500 *Apply/Instantiate Safety Pattern.* The Monitor-Actuator Pattern was instantiated as depicted in Figure 4. Only changes to the BMS component were made.

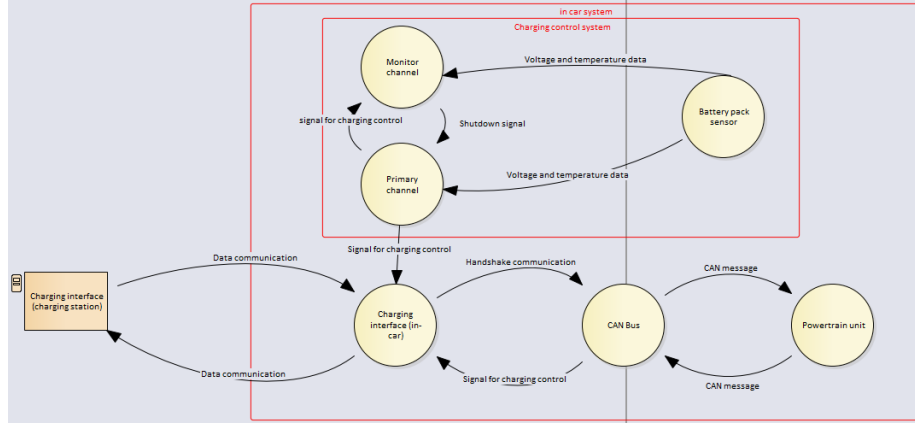


Figure 5: Threat modeling of architecture (Tool: Enterprise Architect 14 with ThreatGet Plugin [31])

4.2.2. Security Pattern Engineering

Perform Security Analysis. In this context, Security Engineering follows the initial definition of a safety pattern to identify potential security threats and vulnerabilities in the design, and to assess the risks in order to find appropriate countermeasures and apply corresponding security patterns. In this example, we use the threat modeling methodology [32], in which a system is modeled in a Data Flow Diagram (DFD). When modeling the functional blocks from the safety pattern (cf. Figure 4) 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 (CI) is split into two

520 parts: an in-car CI 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
525 components beyond the scope of the safety pattern. Since the communication between the primary and monitor channel and the CI goes through the CAN bus, and the powertrain unit is connected to the same bus, the security of the CI also influences the security of the powertrain unit. Thus even though the two safety modules cannot be attacked directly due to the unidirectional data
530 flows, there are risks that an attacker might use the system charging function to attack the powertrain unit. Figure 5 shows the modeled architecture in DFD using the free cyber-security modelling addin for Enterprise Architect [31].

Asset: BMS software and its related functions and messages on the CAN bus are the main assets

535 **Threat:** Attack CAN bus through in-car CI

Comment: The communications from the external CI to the CAN bus is responsible for establishing and maintaining communications for charging control. An attacker can use the in-car CI as an entry point by compromising the external CI or tampering with the communications between the interfaces to inject
540 malicious content into the CAN bus.

Impact: Safety impact of the identified threats on the assets.

Comment: The safety analysis defines the initial scope of the security analysis. The impact analysis links the results from safety and security analysis. Table 1 gives an example of the outcome of the security impact analysis. The co-analysis
545 provides grounds for detailed security engineering requirements on the system.

Security requirements: The security requirements are related to authentication, authorization, non-repudiation, accountability, data integrity, confidentiality, privacy, and availability, which need to be further allocated to system and component level during the design. In this example, we skip the assessment
550 of severity of the impact so we consider all safety impacts when deriving the

Table 1: Security impact analysis

Asset	Security impact	Safety Impact
BMS soft-ware	Integrity of BMS soft-ware	Overcharging battery system
CAN bus communication	Availability of CAN messages to their intended ECUs	Disturbing ECU functioning on the power-train system

security requirements.

Select Security Pattern. During security design, other factors beside requirements such as available solutions, performance and cost need to be taken into consideration. Since current CAN does not support encrypted and authenticated communication, one possible solution is to add a security gateway between the external unit and the internal CAN bus as shown in Figure 6. The security gateway is the application of the security pattern “firewall” that is placed between an unprotected internal network and untrusted external entities when communication to the outside is inevitable (see Figure A.11). The firewall pattern is a basic security measure that controls incoming and outgoing network connections between a protected and an untrusted network. It provides network access control that restricts which hosts can be accessed on the internal network. Note that other security patterns can be applied as well. For example, we might apply the “security proxy” pattern, in which an entity on the communication path between the charging station and the BMS software will not only translate different communication protocols (e.g. Ethernet to CAN), but also authenticate the charging station and verify the validity of the communication. As a repeatable solution, the security gateway is not limited to the CI. It can be applied to any communication between the CAN bus and untrusted external devices. In general, the gateway controls the network access to the internal ECUs according to predefined security policies and can also inspect

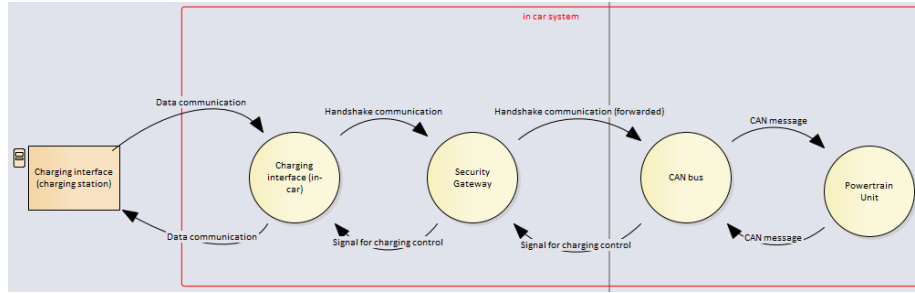


Figure 6: Security Gateway as a security pattern (Tool: Enterprise Architect 14 with Threat-Get Plugin [31])

packet content to detect intrusion attempts and anomalies. In this way, it adds security protection and segments the system without fundamentally changing the existing in-car system architecture.

575 *Select/Instantiate Security Pattern.* In Figure 6, we see the altered architecture with the Security Gateway module.

4.2.3. Safety and Security Co-Engineering Loop

Beyond the many benefits, a security gateway might introduce latency into the communication or block critical messages due to false-positives, which is a
580 subject of safety impact analysis.

Safety Impact Analysis. With the inputs from previous tasks we perform a HAZard and Operability Study (HAZOP) analysis to identify potential anomalies in the provision of the service controlling the CI (cf. Table 2). The focus is thus on the changes performed to the architecture by the security engineers.

585 Based on the analysis we identified failure modes Omission and Late as potential causes of a hazard (cf. Table 2). 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 CI and the BMS. This small delay can cause
590 a minor amount of extra charging in the battery which is not a source of hazard and need no further investigation in the safety and security co-engineering loop.

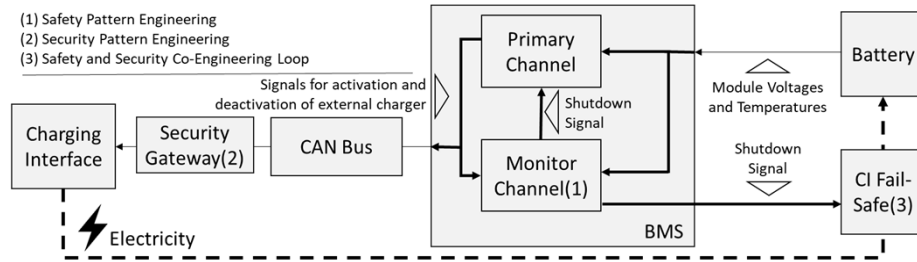


Figure 7: Architecture after the first iteration of safety and security co-engineering

Table 2: HAZOP Guideword analysis of the architecture

Guideword	Possible causes	Possible consequences
Commission	-	-
Omission	Gateway blocks message to stop charging - Message gets corrupted	Charging Interface keeps providing energy to battery
Early	-	-
Late	Additional processing time slows down reaction time of components	Late disconnection of battery
Value High	-	-
Value Low	-	-

From the input received from the previous phase, we also discovered that the safety functions on the CI will not suffice in the case of a hacker attack. This need a further investigation by performing a Safety and Security Analysis.

595 *Safety and Security Co-Analysis.* The two-stage SAHARA method then combines the outcome of the security analysis with the outcomes of the safety analysis. The SAHARA method applies a key concept of the HARA approach, the definition of ASILs, to the STRIDE analysis outcomes. Security threats that might lead to a violation of safety goals can be handed over to HARA for further safety analysis. This helps to improve completeness of safety analysis in terms of the safety requirement of analysis of 'foreseeable misuse', in this case hazardous events initiated due to security attacks. For the battery system it can be seen that security hazard are aiming a overloading of the battery because the CI keeps providing energy to the battery.

605 *Harmonize.* To tackle this issue a CI fail-safe device connected to the Monitor channel was integrated (cf. Figure 7). Of course, one obvious drawback in this solution is the extra cost incurred due to extra hardware and installation. The changes in the architecture neither create new vulnerabilities nor jeopardize the current mechanisms already in place. Furthermore, additional requirements can be added to the Security Gateway in the security concept, specifying that firewall rules shall not block critical messages in charging process, which can be implemented and tested in the development phase.

Trade-Off. 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. In [33] we presented a quantitative methodology for security risk assessment by combining FMVEA and SAHARA methods with the FAIR method. This enables the analysis of security and failure event chains, as well as a coordinated risk management for safety and

security features. In this case, 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
625 the safety pattern directly.

Argumentation. During the task argumentation the safety and security concept is finally available and all argument fragments through the overall workflow are consolidated: GSN argumentation fragments of the chosen patterns (e.g. security pattern for firewall in Figure A.11), co-analysis results, and evidences
630 of the elaborated safety and security measures from harmonization and/or trade-off. The joint assurance case contains the combined safety and security case.

4.2.4. Modeling of the System

For the item definition of the system we used the block definition diagram (bdd) of SysML. The blocks, their hierarchy and multiplicities (e.g. exactly one
635 module consists of at least one cell) are defined based on the use case description and requirements (cf. Figure 8 [Top]). Furthermore, all input and output ports are defined as known so far. This item definition is compliant to ISO 26262 and subject to further safety and security analyses.

4.2.5. Modeling Patterns

In a similar way, patterns are modeled. Patterns may contain UML classes
640 (or blocks in SysML, as equivalent) and their relationships (cf. Figure 8 [Bottom]). The diagram is saved as a pattern and stored in a separate XML file format and stored in the pattern repository. A specific save dialog offers several options (e.g. specification of name, file-name, and category). The classes
645 of the patterns may be replaced with classes from your system model during the integration later on, which needs to be enabled by ticking the boxes in the corresponding columns (create/merge/instance/type). To give an example, the *merge* option replaces an existing model element with an element defined in

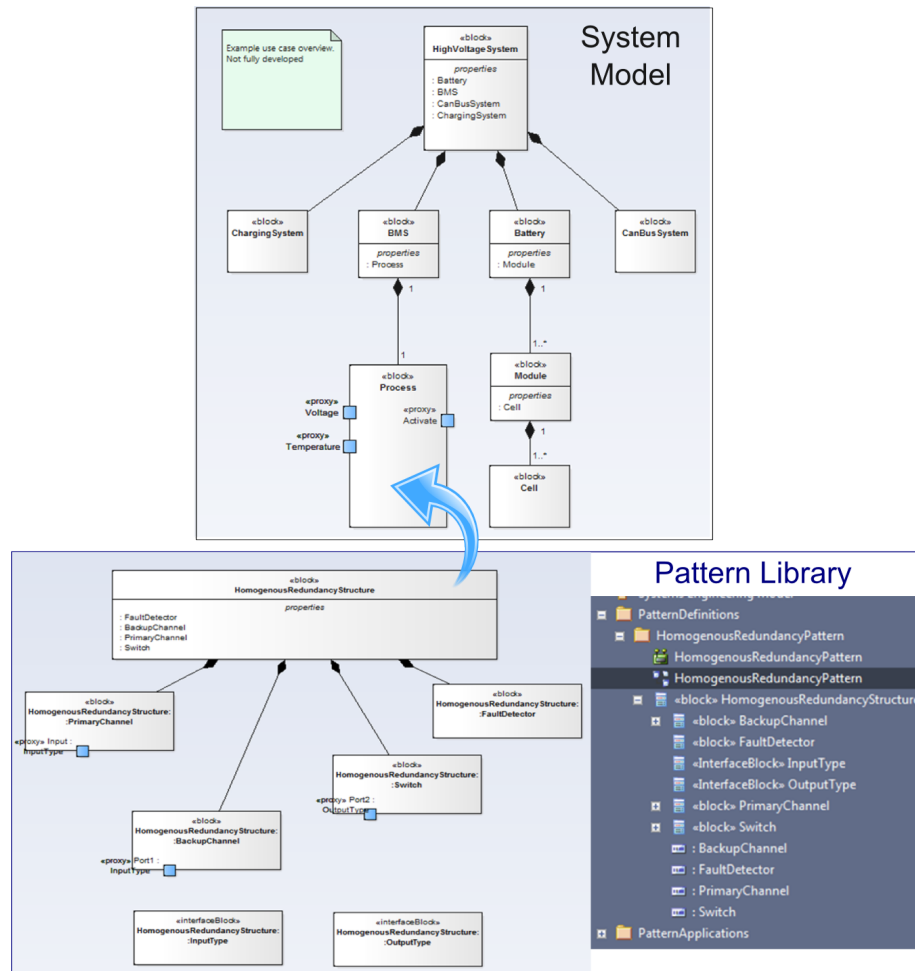


Figure 8: [Top]: Item Definition in SysML | [Bottom]: Creation of Safety Pattern [Tool:Enterprise Architect]

the pattern. This is especially useful for larger patterns, and ensures seamless
650 integration of the pattern with an existing model.

4.2.6. *Pattern Instantiation*

The pattern may then be applied to the system model by drag and drop of
the pattern to the diagram by a following dialog window, where it is possible
to assign existing elements to the pattern (cf. Figure 8 Bottom). In the exam-
655 ple the Pattern “Primary channel” is mapped to the BMS “Process” and the
Pattern “HomogenousRedundancyStructure” is mapped to the BMS. Based on
the mapping the tool updates the bdd with the missing classes from the chosen
pattern and the pattern import is completed.

4.2.7. *Interaction with Analysis Tool*

660 Threat Modeling is a wide spread security analysis, which is also one of
the contributions for co-analysis techniques like SAHARA or FMVEA. The
main tool for Threat Modeling has been a plug-in for MS Visio, which only
provides a graphical modeling support. The tool from AIT in development
called “ThreatGet” integrates Threat Modeling into Enterprise Architect. With
665 the integration into EA the threat modeling is based on a SysML model and
resulting threats can be integrated into the workflow and guarantee traceability.

4.3. *Lessons Learned*

The definition of patterns in the EA tool is done via simple UML class
diagrams and works for SysML blocks as well. The pattern are saved in separate
670 files in a software versioning and revision control system, which allowed to built
easily a “library of patterns”. Furthermore, it is easy to apply the patterns
to the system model via drag and drop. The pattern mechanism of the EA
tool seems to work at this stage for bdd diagrams only, which means that the
pattern “just reminds you to add needed blocks” and the engineer has to connect
675 them as needed according to the specific context. The concrete interconnections
between blocks, normally realized via interfaces and ports, are out of scope of
the EA patterns. This is due to the fact that at this point in time the exact

definition of ports and interfaces, etc. are unknown. The described approach using EA limits the application of patterns to early development phases.

680 4.4. Industry Relevance

Currently vehicle chargers can be mainly classified into: AC charging that uses residential power outlet, DC fast charging (DCFC) that charges a vehicle by DC in much shorter time, and inductive charging that eliminate the use of cables. All of them require certain message exchange between the charging station
685 and the vehicle during the charging process to regulate the current provided to the vehicle. SAE recommendations are often used in the industry to implement such communications on top of the physical and data link layer. For example, SAE J2847/2 “Communication between plug-in vehicles and off-board DC chargers” [34] defines message format for DC charging. These messages might be
690 manipulated to sabotage the charging process that leads to battery explosion. Therefore, in SAE J2931/7 “Security for plug-in electric vehicle communications” [35] a set of security requirements are specified for different stakeholders involved in the system. On the other hand, standards such as ISO 17409:2015 “Preview Electrically propelled road vehicles - Connection to an external electric power supply - Safety requirements” [36] specifies the safety requirements.
695 The safety and security pattern described here has been instantiated in certain degree in actual vehicles, which use a gateway to avoid direct data connection to charging station and relays controlled by BMS to avoid overcharging. Therefore, safety and security patterns can be an repeatable and efficient way to address
700 and enforce safety and security requirements from related standards in system development based on industry best practices.

5. Discussion and Conclusion

This paper focused on the selection, combination, and application of safety and security patterns for automotive system engineering. The introduction of
705 the Combined Pattern-based Engineering Workflow provides a systematic and

integrated way of safety- and security-related pattern engineering. It provides comprehensive guidance and packaged knowledge with respect to the integration, as well as auxiliary existing work products, such as the (initial) results of safety and security analyses and argumentation fragments. The availability
710 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. The Safety and Security Co-Engineering Loops as well as the correlated guidance provided by our augmented pattern descriptions help to align the required activities systematically
715 and foster a tight integration of safety- and security-related process steps. For instance, this means if a safety (architectural) pattern will be selected to address a specific safety requirement, additional information and guidance with respect to correlated core aspect from a security point of view are provided. A security pattern, in turn, can have a safety impact, which is again explicitly specified.

720 With the presented approach, the decision which pattern fits best for a specific system can be done more systematically (particularly by non-expert engineers) and taking into account the problem to be addressed and the context of the system. In general, safety and security engineering are very closely related disciplines and their synergy can be fostered when their similarities are
725 recognized and adequate interactions are established correctly.

To demonstrate the benefits of the presented approach an automotive case study demonstrated the practical realization of our approach: Architecture of an automotive battery system was described in a semi-formal way, including identification of its main components, physical interconnections, and flows of in-
730 formation. The use case has been reduced in complexity for illustration purposes and it highlights the benefits of applying the Pattern Engineering Workflow for the concept phase development.

With the presented approach, we aimed to benefit from the typical strong points of patterns to overcome the lack of general security knowledge as well as
735 organizational weak points (i.e. no integrated safety-security teams, distributed or lacking security knowledge etc) that are prevalent in the automotive domain.

It is a promising approach that should help accelerating the application of adequate safety and security co-engineering in industry. In particular, we believe the type of pattern we introduced constitute a way to remediate the lack of security knowledge and facilitate easier and more informed integration of these two separate yet interfering disciplines. Additionally, initial positive experiences were shown, which were gained by a tool supported model-based systems engineering application, where patterns were created and applied for a specific automotive use case scenario.

Acknowledgment

Dedicated to our co-author late Christian Kreiner, who was impressive for many reasons and has been a wonderful teacher, co-worker, leader and friend. You have been everything one could look for in a good mentor, the true Master Yoda, and made working with you an exciting, inspiring and memorable experience. We will always be grateful to you for your support and kindness. May the force still be with you.

Research leading to these results has received funding from the EU ECSEL Joint Undertaking under grant agreement no. 692474 (project AMASS), EU ECSEL JU project SCOTT (no. 737422), EU Horizon 2020 research and innovation programme under grant agreement no. 732242 (project DEIS), 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.

List of Abbreviation

Table 3: List of Abbreviation

<i>Abbrev.</i>	<i>Definition</i>	<i>Abbrev.</i>	<i>Definition</i>
AC	Alternating Current	HARA	Hazard Analysis and Risk Management
ASIL	Automotive Safety Integrity Level	HAZOP	HAZard and OPerability Study
BMS	Battery Management System	HV	High Voltage
C	Controlability	ID	Identifier
CAN	Controller Area Network	ISO	International Organization for Standardization
CI	Charging Interface	IT	Information Technology
CPS	Cyber-Physical System	MBSE	Model-Based Systems Engineering
DCFC	Direct Current Fast Charging	MS	MicroSoft
DFD	Data Flow Diagram	S	Severity
E	Exposure	SAE	Society of Automotive Engineers
E/E	Electric and/or Electronic	SAHARA	Security Aware Hazard Analysis and Risk Assessment
EA	Enterprise Architect	STRIDE	Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service, Elevation
ECU	Electronic Control Unit	SysML	System Modelling Language
FDIS	Final Draft International Standard	TARA	Threat Analysis and Risk Assessment
FMVEA	Failure Mode and Vulnerability Eect Analysis	UML	Unified Modelling Language
GSN	Goal Structuring Notation	XML	Extensible Markup Language

References

- [1] A. Joshi, M. P. Heimdahl, S. P. Miller, M. W. Whalen, Model-Based Safety Analysis, Tech. Rep. CR-2006-213953, NASA (2006).
- [2] B. Kaiser, V. Klaas, S. Schulz, C. Herbst, P. Lascych, Integrating system modelling with safety activities, in: SAFECOMP’10 Proceedings of the 29th international conference on Computer safety, reliability, and security, Springer, 2010, pp. 452–465.
- URL http://link.springer.com/chapter/10.1007/978-3-642-15651-9_33
- [3] T. Amorim, H. Martin, Z. Ma, C. Schmittner, D. Schneider, G. Macher, B. Winkler, M. Krammer, C. Kreiner, Systematic pattern approach for safety and security co-engineering in the automotive domain, in: International Conference on Computer Safety, Reliability, and Security, Springer, 2017, pp. 329–342. doi:10.1007/978-3-319-66266-4_22.

- [4] ISO, ISO 26262 Road vehicles – Functional safety (2011).
- [5] SAE, SAE J3061 Cybersecurity Guidebook for Cyber-Physical Vehicle Systems (2016).
- 780 [6] C. Schmittner, G. Griessnig, Z. Ma, Status of the development of iso/sae 21434, in: European Conference on Software Process Improvement, Springer, 2018, pp. 504–513.
- [7] G. Macher, E. Armengaud, C. Kreiner, E. Brenner, C. Schmittner, Z. Ma, H. Martin, M. Krammer, Integration of security in the development life-
785 cycle of dependable automotive cps, in: Handbook of Research for Cyber-Physical Systems Ubiquity, IGI Global, 2017.
- [8] C. Schmittner, Z. Ma, E. Schoitsch, T. Gruber, A case study of fmvea and chassis as safety and security co-analysis method for automotive cyber-physical systems, in: Proceedings of the 1st ACM Workshop on Cyber-Physical System Security, CPSS '15, ACM, New York, NY, USA, 2015, pp. 69–80. doi:10.1145/2732198.2732204.
790
- [9] G. Macher, H. Sporer, R. Berlach, E. Armengaud, C. Kreiner, Sahara: A security-aware hazard and risk analysis method, in: 2015 Design, Automation Test in Europe Conference Exhibition (DATE), 2015, pp. 621–624. doi:10.7873/DATE.2015.0622.
795
- [10] C. Schmittner, Z. Ma, T. Gruber, E. Schoitsch, Safety and security co-engineering of connected, intelligent, and automated vehicles, ERCIM News 109 (2017) 23–24.
- [11] T. Gruber, C. Schmittner, M. Matschnig, B. Fischer, Co-engineering-in-the-loop, Computer Safety, Reliability, and Security (2018) 151.
800
- [12] C. Alexander, S. Ishikawa, M. Silverstein, M. Jacobson, I. Fiksdahl-King, S. Angel, A pattern language (1977).

- [13] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns: Elements of Reusable Object-oriented Software, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995.
- [14] A. Armoush, Design patterns for safety-critical embedded systems., Ph.D. thesis, RWTH Aachen University (2010).
- [15] C. Preschern, N. Kajtazovic, C. Kreiner, Building a safety architecture pattern system, in: Proceedings of the 18th European Conference on Pattern Languages of Program, EuroPLoP '13, ACM, New York, NY, USA, 2015, pp. 17:1–17:55. doi:10.1145/2739011.2739028. URL <http://doi.acm.org/10.1145/2739011.2739028>
- [16] B. P. Douglass, Real-Time Design Patterns: Robust Scalable Architecture for Real-Time Systems, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.
- [17] B. P. Douglass, Design Patterns for Embedded Systems in C: An Embedded Software Engineering Toolkit, 1st Edition, Newnes, Newton, MA, USA, 2010.
- [18] L. L. Pullum, Software Fault Tolerance Techniques and Implementation, Artech House, Inc., Norwood, MA, USA, 2001.
- [19] M. Schumacher, Security engineering with patterns: origins, theoretical models, and new applications, Vol. 2754, Springer Science & Business Media, 2003.
- [20] N. A. Delessy, E. B. Fernandez, A pattern-driven security process for soa applications, in: 2008 Third International Conference on Availability, Reliability and Security, 2008, pp. 416–421. doi:10.1109/ARES.2008.89.
- [21] N. E. Petroulakis, G. Spanoudakis, I. G. Askoxylakis, A. Miaoudakis, A. Traganitis, A pattern-based approach for designing reliable cyber-physical systems, in: 2015 IEEE Global Communications Conference (GLOBECOM), 2015, pp. 1–6. doi:10.1109/GLOCOM.2015.7417794.

- [22] J. A. Estefan, et al., Survey of model-based systems engineering (mbse) methodologies, IncoSE MBSE Focus Group 25 (8) (2007) 1–12.
- [23] OMG Systems Modeling Language (OMG SysML) (2012), <http://www.omg.org/spec/SysML/1.3/> (Jun. 2012).
- 835 [24] S. Friedenthal, A. Moore, R. Steiner, A practical guide to SysML: The systems modeling language, Morgan Kaufmann, 2014.
- [25] G. Biggs, T. Sakamoto, T. Kotoku, A profile and tool for modelling safety information with design information in SysML, *Software & Systems Modeling* 15 (1) (2016) 147–178. doi:10.1007/s10270-014-0400-x.
840 URL <http://link.springer.com/10.1007/s10270-014-0400-x>
- [26] Muhammad Sabir Idrees, Yves Roudier, Ludovic Apvrille, A Framework Towards the Efficient Identification and Modeling of Security Requirements, 2010.
- [27] C. Preschern, N. Kajtazovic, C. Kreiner, Security analysis of safety patterns, in: *Proceedings of the 20th Conference on Pattern Languages of Programs*, The Hillside Group, 2013, p. 12.
845
- [28] A. Karahasanovic, P. Kleberger, M. Almgren, Adapting threat modeling methods for the automotive industry, in: *Proceedings of the 15th ESCAR Conference*, 2017, pp. 1–10.
- 850 [29] Z. Ma, C. Schmittner, Threat modeling for automotive security analysis, *Advanced Science and Technology Letters* 139 (2016) 333–339.
- [30] M. Hamad, M. Nolte, V. Prevelakis, Towards comprehensive threat modeling for vehicles, in: *the 1st Workshop on Security and Dependability of Critical Embedded Real-Time Systems*, 2016, p. 31.
- 855 [31] Sparx Services CE – Cyber Security Modeling – Security by design, [Online; accessed 5. Jul. 2019] (Jul 2019).
URL <https://cybersecurity.sparxservices.eu>

- [32] A. Shostack, Threat Modeling: Designing for Security, Wiley, 2014.
URL <https://books.google.de/books?id=asPDagAAQBAJ>
- 860 [33] J. Dobaj, C. Schmittner, M. Krisper, G. Macher, Towards Integrated Quantitative Security and Safety Risk Assessment, in: Computer Safety, Reliability and Security - SAFECOMP 2019 Workshops - ASSURE, DECSoS, SASSUR, STRIVE, and WAISE , Vol. LNCS of Lecture Notes in Computer Science, Springer International Publishing AG, 2019.
- 865 [34] SAE, SAE J2847/2 Communication between plug-in vehicles and off-board DC chargers (2015).
- [35] SAE, SAE J2931/7 Security for Plug-In Electric Vehicle Communications (2018).
- 870 [36] ISO, ISO 17409 Electrically propelled road vehicles – Connection to an external electric power supply – Safety requirements (2015).

Appendix A. Pattern Examples

This section shows some pattern examples of our approach to describe pattern:

- pattern template see Figure A.9;
- safety pattern example see Figure A.10;
- 875 • security pattern example see Figure A.11.

PATTERN TEMPLATE	
Section	<i>Description (What it contains and where the contents comes from)</i>
Pattern	Name The pattern name is taken from the existing pattern - most of which come from [Armoush, 2010].
Pattern Type	Classification into hardware/software and fail-safe/fail-over. This classification comes from the pattern types which we classify during the process of building up the pattern language.
Also Known	As Other names for the pattern used in literature.
Context	The contents of this section comes from existing patterns and was structurally adapted to fit our pattern system.
Problem	The contents of this section comes from existing patterns and was structurally adapted to fit our pattern system.
Forces	The contents of this section comes from existing patterns (mostly from [Grunske, 2003]) and was structurally adapted to fit our pattern system.
Solution	The solution is shortly described in a few sentences and the structure of the safety architecture is shown in a diagram. Most of the diagrams are based on [Armoush, 2010] and [Douglass, 2002].
GSN Diagram	<p>This section contains a Goal Structuring Notation (GSN) diagram which relates the main aim of the pattern to the architectural design decisions which were taken to achieve this aim. GSN is a graphical notation which is often used in the safety domain to describe how a certain goal is achieved. The advantage of using this notation is that it is familiar to safety experts and the resulting pattern GSN diagram can be used to structurally argue about a system's safety. Figure 1 shows the basic elements of GSN and explains them. The GSN diagram is based on information about the usage of basic architectural design decisions (architectural tactics) which are applied in the pattern. We obtain these tactics from pattern descriptions according to a method presented by [Kumar and Prabhakar, 2010] which we will cover in more detail in the next section.</p> <p>Fig. 1. GSN concepts used in this paper taken from [GSN Working Group 2011]</p>
Consequences	The consequences are split into a part containing general consequences and a part explicitly covering quality attribute related consequences (e.g. consequences on safety or availability). The information about the consequences mostly comes from the safety patterns from [Armoush, 2010] and [Grunske, 2003].
General Scenarios	This section contains scenarios of the system which can, for example, be used during architecture evaluations. The scenarios are mined from patterns as suggested in [Babar, 2007] by manually searching the problem and solution statements for scenarios for relevant quality attributes (in our case focused on safety). Scenarios are included in the patterns because there are existing safety reasoning frameworks which are based on scenarios ([Wu, 2007]) and the information of the scenarios is also needed to build up our GSN diagrams.
Known Uses	This section presents known uses for the patterns. We added this information by searching for literature which applies the pattern. We just included patterns for which we could find at least three known uses.
Credits	References to previous work on the pattern.
References	<p>[Armoush, 2010] ARMOUSH, A. 2010. Design patterns for safety-critical embedded systems. Ph.D. thesis, RWTH Aachen University.</p> <p>[Douglass, 2002] DOUGLASS, B. P. 2002. Real-Time Design Patterns: Robust Scalable Architecture for Real-Time Systems. Pearson.</p> <p>[Grunske, 2003] GRUNSKKE, L. 2003. Transformational Patterns for the Improvement of Safety Properties in Architectural Specification. In Proceedings of The Second Nordic Conference on Pattern Languages of Programs (VikingPLoP).</p> <p>[Wu, 2007] WU, Weihang (2007). Architectural Reasoning for Safety- Critical Software Applications. PhD thesis. University of York.</p> <p>[Babar, 2007] BABAR, M.A. (2007). Improving the Reuse of Pattern-Based Knowledge in Software Architecting. In: EuroPLoP. Lero, Ireland, 7–11.</p> <p>[Kumar and Prabhakar, 2010] KUMAR, Kiran and T.V. PRABHAKAR (2010b). Pattern-oriented Knowledge Model for Architecture Design. In: 17th Conference on Pattern Languages of Programs (PLOP).</p> <p>[GSN Working Group 2011] GSN WORKING GROUP. 2011. GSN Community Standard Version 1. http://www.goalstructuringnotation.info/.</p>

Figure A.9: Pattern Template

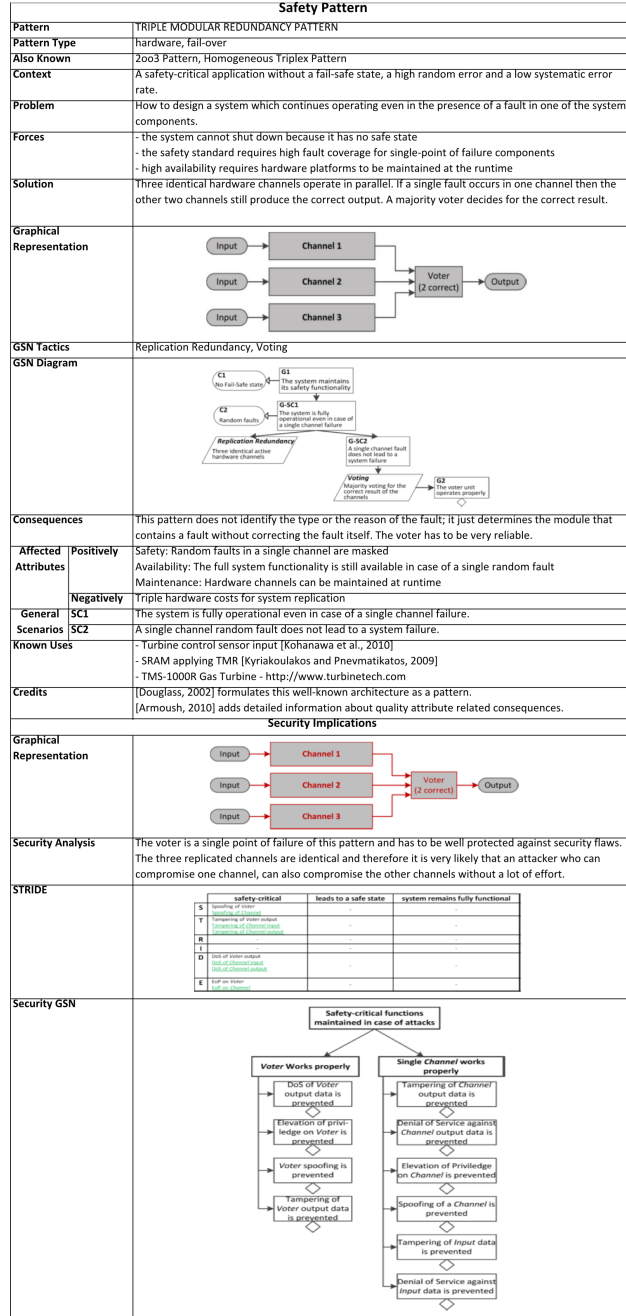


Figure A.10: Safety Pattern Example

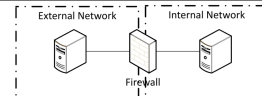
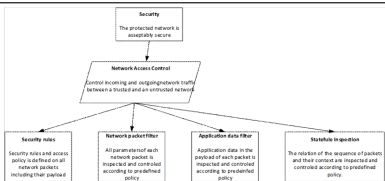

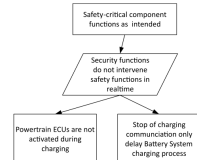
Security Pattern		
Pattern	FIREWALL PATTERN	
Pattern Type	Hardware and Software	
Also Known	Network access control	
Context	Placed in between the incoming and outgoing communication of the internal and external network	
Problem	The need to control incoming and outgoing network traffic according to predefined security rules. The external network is assumed to be untrusted. The internal network is assumed to have weak or no security protection. Therefore, it needs perimeter security protection and monitoring.	
Forces	Defense in depth requires that layers of security mechanisms are placed to protect assets	
Solution	A hardware with two network interfaces connect the internal and external network, respectively. A software with a higher privilege runs on the hardware that virtually separate the communication on the network layer. The software inspect the network packet and its payload and make decisions whether to forward or drop the packet from one interface to another interface, according to a predefined policy (in whitelist or blacklist). More advanced firewall can make ad-hoc decisions based on the context of the network traffic.	
Graphical Representation		
GSN Tactics	Defense node that stop	
GSN Diagram		
Consequences	Firewall is a programmed network traffic controller. It must allow certain traffic to go through while blocking other that deemed malicious. As the decision is purely based on the security rules or firewall policy, it might have false-positive or false negative. The consequence of false-positive is that certain network packets will be dropped, and the consequence of false-negative is that malicious network packets will get through the firewall which subvert the security protection.	
Affected Attributes	Positively	Confidentiality: encryption can be performed on firewall on the traffic originated from external network and ended at the firewall. Integrity: authentication and authorization can be performed on the end point on teh firewall. For example, a TLS tunnel can be established on a host on the external network to the firewall to establish a secure communication channel.
	Negatively	Availability: false positive will block legitimate network traffic, which cause availability issue.
General	SC1	All network traffic goes through the firewall
Scenarios	SC2	The firewall needs to be configured properly in order to function
Known Uses	Common security control in most enterprise networks.	
Credits	[Security pattern catalogue https://people.cs.kuleuven.be/~koen.yskout/icse15/catalog.pdf]	
Safety Implications		
Graphical Representation		
Safety Analysis	HAZOP	
Provision	Omission	Firewall rules block safety-critical communication
	Commission	NA
Timing	Early	NA
	Late	Firewall delays safety-critical communication, leading to a violation of real time behaviour
Value	Subtle	Encryption at the firewall (failure in Firewall) leads to an change in submitted value
	Coarse	NA
Safety GSN		

Figure A.11: Security Pattern Example