Deriving Safety Case Fragments for Assessing MBASafe's Compliance with EN 50128

Barbara Gallina[1], Elena Gómez-Martínez[2], and Clara Benac Earle[3]

[1] Mälardalen University, Västerås, Sweden
barbara.gallina@mdh.se
[2] University of East London, London, UK
e.gomez@uel.ac.uk
[3] Universidad Politécnica de Madrid, Madrid, Spain
cbenac@fi.upm.es

Abstract. According to EN 50129, manufacturers of rail vehicles shall justify via a safety case that their vehicles are adequately safe for their intended applications. MBASafe is a recently proposed and potentially innovative design and verification process. In the presence of compelling arguments concerning its adequacy as process evidence, MBASafe could support the safety claims within the required safety cases. In this paper, we contribute to partially justify the adequacy of MBASafe to act as process evidence. To do that, we first manually check if MBASafe includes EN 50128-compliant process elements, then we model MBASafe in compliance with Software Process Engineering Meta-model 2.0, then, we derive process-based arguments from the MBASafe process model by using MDSafeCer, the recently introduced Model Driven Safety Certification method. By doing so, we provide a twofold contribution: we further validate MDSafeCer in the rail domain and we strengthen MBASafe.

Keywords: EN 5012x, model-driven safety certification, process assessment.

1 Introduction

According to the CENELEC standard series, manufacturers of rail vehicles shall justify via a safety case that their vehicles are adequately safe for their intended applications. More specifically, the CENELEC EN 50129-compliant safety case should include arguments aimed at explaining why the included evidence (e. g., safety and quality management) is adequate to support the safety claims. Arguments should specifically refer to the appropriate Safety Integrity Level (SIL) since the stringency from one level to another changes. Recently proposed and potentially innovative engineering methods could act as process-related evidence. However, to ease their acceptance within the rail industrial settings, the adequacy of these methods need to be justified. MBASafe [1] is a recently proposed and potentially innovative model-driven process for the design and verification of software architectures. MBASafe has been validated in research settings in cooperation with industry [1]. The adoption of MBASafe in the rail domain, however, is not straightforward due to the current absence of compelling arguments

concerning its adequacy, i.e., arguments aimed at explaining that the selection of process elements that composes MBASafe, aimed at guiding the design of rail vehicles-related subsystems, is compliant with the CENELEC requirements. MDSafer is a method aimed at speeding up the creation of process-based arguments, derived from process models, given in standardized process languages e.g., SPEM (Software Process Engineering Meta-model) 2.0 [2]. The usage and potential effectiveness of MDSafer has been illustrated in the automotive [3] and rail domain [4]. However, no in-depth illustration has been attempted so far. In this paper, we use MDSafeCer to derive part of the needed justification concerning the adequacy of MBASafe as safety and quality management evidence. By doing so we provide a twofold contribution: we further extend and validate MDSafeCer and we strengthen MBASafe by deriving safety case fragments aimed at showing its adequacy to design software sub-systems in compliance with EN 50128. More specifically, we consider the design of a door control management subsystem (within a specific train control monitoring system) in a suburban train. This subsystem is expected to have doors with a button that enables passengers to open them upon request. The malfunctioning of this system may endanger the system safety. The assumed Safety Integrity Level (SIL) is SIL 2. Given this system, we focus our attention on justifying adequacy with respect to SIL 2. Given the pattern-based nature of our justification, it can be flexibly changed to argue about a different level, where necessary.

The rest of the paper is organized as follows. In Section 2, we present essential background. In Section 3, we collect elements of EN 50128-compliance and we model in SPEM2.0 the compliant portion of MBASafe. In Section 4, we derive safety case fragments for arguing that MBASafe partially meets EN 50128. In Section 5, we discuss related work. Finally, concluding remarks and future work can be found in Section 6.

2 Background

In this section we present the essential background on which we base our work.

2.1 Safety Cases and Safety Cases Representation

A safety case is defined as "a structured argument, supported by a body of evidence, that provides a compelling, comprehensible and valid case that a system is safe for a given application in a given environment" [5]. Such argument typically includes process and product-based sub-arguments. To document safety cases, several approaches exist. GSN [6] is one of them and it is here selected because of its active community and its current level of maturity. GSN is a graphical notation, which permits users to structure their argumentation into flat or hierarchically nested graphs (constituted of a set of nodes and a set of edges), called goal structures. To make the paper self-contained, in Fig. 1, we recall a subset of the GSN concrete syntax used in Section 4. As Fig. 1 shows, all the nodes are characterized by an identifier (ID) and a statement, which is supposed to be written in natural language.

We recall that a *Goal* represents a claim about the system; a *Strategy* represents a method that is used to decompose a goal into sub goals; a *Solution* represents the evidence that a particular goal has been achieved; a *Context* represents the domain or scope in which a goal, evidence or strategy is given; *Supported by* represents an inferential (inference between goals) or evidential (link between a goal and the evidence used to substantiate it) relationship. Finally, *In context of* represents a contextual relationship. To create argumentation patterns, i.e., reusable goal structures, specific pattern constructs are at disposal, as shown in Fig. 1. Within patterns, in addition to the constructs presented in Fig. 1, curly brackets are also used to denote variables. SACM (Structured Assurance Case Metamodel) [7] is an OMG standard aimed at unify and standardize the graphical notations (including GSN) broadly used for documenting safety cases. At the time being SACM only addresses a subset of GSN modeling elements. Pattern constructs, for instance, are not addressed yet.

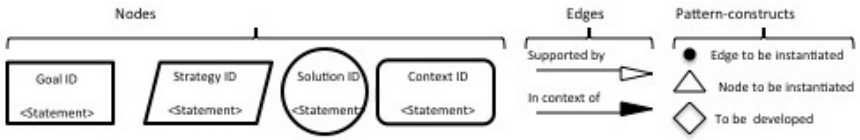


Fig. 1. Subset of GSN concrete syntax.

2.2 The CENELEC EN 5012x

The CENELEC EN 5012x is a family of standards that contains requirements and recommendations concerning processes to be followed for the development and assurance of safety-critical systems. This family of standards is used for the certification of railway systems and signaling control-command equipment. As it was documented within the deliverable D6.1 of the MODSafe project [8], Light Rail, Metros, Trams are still characterized by a diversified landscape of safety requirements, safety models, roles and responsibilities, safety approval, acceptance and certification schemes. However, convergence towards the CENELEC standard series is evident. In this section, we briefly present the portions of EN 50126, EN 50129 and EN 50128 that are necessary to understand Section 4.

EN 50126 [9] defines a fourteen-phase process to manage Reliability, Availability, Maintainability and Safety at system level. The Risk Analysis Phase is the third phase. The objective of this phase is multi-fold: 1) identification of the hazards associated with the system; 2) estimation of the risk associated with the hazards; 3) development of a process for risk management. One of the outcome of the Risk Analysis phase is the assignment of a SIL to any safety relevant function or system or sub-system or component. A SIL specifies a target level of risk

reduction and is typically defined in components that operate in a safety-critical system. There are four discrete integrity levels associated with SIL with SIL 4 the most dependable and SIL 1 the least. The SIL allocation is made taking into account the rate of dangerous failures and tolerable hazard rate of the function, system, sub-system or component. The SIL of a system to be developed is determined on system level. The software "inherits" the SIL as any other part of the system through decomposition. Then, EN 50128 defines what must be done to develop SW functions with that SIL.

EN 50129 [10] defines the conditions that shall be satisfied in order that a safety-related electronic railway system/sub-system/equipment can be accepted as adequately safe for its intended application. These conditions are constituted of three types of evidence: Evidence of quality management, Evidence of safety management, and Evidence of functional and technical safety. The documentary evidence that these conditions have been satisfied shall be included in a structured safety justification document, known as the safety case. The safety case shall be structured in six parts. In this sub subsection we limit our attention to the following parts: Part 2 Quality Management Report, this shall contain the evidence of quality management, e.g., evidence of adequate organizational structures as well as evidence of adequate personnel competence and training; Part 3 Safety Management Report, this shall contain the evidence of safety management, e.g., evidence that the safety management process consists of a number of phases and activities, which are linked to form the safety life-cycle in compliance with EN 50126 and with EN 50128 at software sub-system level. The software architecture design phase should for instance be aligned with the system architecture design. Part 6 Conclusion, this shall summarize the evidence presented in the previous parts of the safety case, and argue that the relevant system/sub-system/equipment is adequately safe, subject to compliance with the specified application conditions.

It should be noted that the depth of the evidence presented and the extent of the supporting documentation should be appropriate to the SIL of the system/sub-system/equipment under scrutiny.

EN 50128 [11] focuses on processes for the development, deployment and maintenance of safety-related software for railway control and protection applications. EN 50128 does not mandate the use of a particular software development lifecycle. It only provides normative tables and recommendations concerning specific process elements, e.g., roles, work products, techniques, tools, tasks. Illustrative software route maps are indicated, however, a process engineer is responsible for the selection and composition of adequate process elements aimed at achieving the required software integrity level. To make the paper self-contained, we recall those process elements related to the Software Architecture & Design Phase that are in relation with MBASafe.

Tasks and related work products- The design task should receive in input the Software Requirements Specification and should deliver in output the Soft-

ware Architecture Specification, the Software Design Specification, the Software Interface Specifications, the Software Integration Test Specification, the Software/Hardware Integration Test Specification, and the Software Architecture and Design Verification Report. The verification task should receive in input all necessary system, hardware and software documentation and should deliver in output a Software Verification Plan a set of Software Verification Report(s), and a Software Quality Assurance Verification Report. The validation task should receive in input all necessary system, hardware and software documentation and should deliver in output a Software Validation Plan, a Software Validation Report and a Software Validation Verification Report.

Guideline- We limit our attention to Annex A. According to Table A.4, formal methods are recommended (R) for SIL 1 and SIL 2 and highly recommended (HR) for SIL 3 and SIL 4. More generally, modeling is HR for SIL1-4. According to Table A.5, formal proofs are R for SIL 1 and SIL 2 and HR for SIL 3 and SIL 4. According to Table A.17, petri nets are R for SIL 1 and SIL 2 and HR for SIL 3 and SIL 4. Finally, according to Table A.22, Object Oriented Detailed Design is R for SIL 1 and SIL 2 and HR for SIL 3 and SIL 4.

Roles- We limit our attention to Annex B. According to Table B.2, a designer shall: transform specified software requirements into acceptable solutions; own the architecture and downstream solutions; define or select the design methods and supporting tools; apply appropriate design principles and standards; develop component specifications where appropriate; maintain traceability to and from the specified software requirements; develop and maintain the design documentation; ensure design documents are under change and configuration control. With respect to expected competencies, a designer shall be competent in: engineering appropriate to the application area, the safety design principles, design analysis & design test methodologies, and understanding the problem domain. Moreover, a designer shall understand: the constraints imposed by the hardware platform, the operating system and the interfacing systems and the relevant parts of EN 50128. Finally, (s)he shall be able to work within design constraints in a given environment.

According to Table B.5, a verifier shall be: competent requirements engineering and experienced in the applications domain and in the safety attributes of the applications domain. Moreover, a verifier shall understand: the overall role of the system and the environment of application; analytical techniques and outcomes; the applicable regulations; and the requirements of EN 50128.

Finally, according to Table B.7, a validator shall be competent in: the domain where validation is carried out as well as various validation approaches / methodologies and be able to identify the most suitable method or combination of methods in a given context. Moreover, he/she shall be: experienced in safety attributes of applications domain; capable of deriving the types of validation evidence required from given specifications bearing in mind the intended application as well as of combining different sources and types of evidence and synthesize an overall view about fitness for purpose or constraints and limitations of the application. A validator shall also have analytical thinking ability and good observation skills as well as overall software understanding and per-

spective including understanding the application environment. Finally, he/she shall understand the requirements of EN 50128. It should be also mentioned that the verifier and validator can be the same person in case of SIL1 and SIL2.

2.3 SPEM 2.0

SPEM 2.0 [2] is the OMG's standard for systems and software process modelling. SPEM 2.0 supports the definition of reusable process content, i.e., work definition elements (e.g., tasks, etc.) as well as elements representing: who is responsible for the work (roles), how the work should be performed (guidance), what should be expected as in/output (work-products) and which tool should be used to perform the work. In Table 1, we recall a subset of SPEM 2.0 modelling elements, which can be interrelated to model static process structures.

Table 1. Subset of SPEM 2.0 modelling elements

Task	Role	WorkProduct	Tool	Guidance
				

2.4 Model-driven Engineering Principles and Derived Methods

Model-driven Engineering (MDE) principles consist of the exploitation of models to capture characteristics at different abstraction levels of the development life-cycle. For automation purposes, vertical as well as horizontal model transformations are used to refine models (model-to-model transformations). A model transformation transforms a source model (compliant with one meta-model) into a target model compliant with the same or a different meta-model. A standard transformation can be defined as a set of rules to map source to the target. Each rule describes how to transform source instances to the identical target.

MBASafe - Gómez-Martínez et al. [1] propose a Model-Based methodology for Assessing (MBA) performance and safety requirements of critical systems at early stages of the design phase. Since this paper is only focused on safety certification, we simplify this methodology taking into account this perspective. We call the simplified methodology MBASafe. The methodology is constituted of four chained tasks, which can be iterated and are: 1) the *design* task (focus on the functional specification) is carried out by the designer and focuses on modeling the software system architecture by means of UML diagrams, being these diagrams the outcome of this step. 2) The non-functional *safety specification* task is carried out by the safety engineer and consists of specifying safety requirements using Safety Contract Fragments (SCF) [12]. SFCs are in turn mapped into OCL constraints and included within the UML diagrams. 3) The

transformation task is aimed at obtaining a formal architectural specification. This activity is carried out by a Petri net expert (Verifier) who translates the UML diagrams augmented with OCL constraints into Generalized Stochastic Petri nets (GSPN) [13]. This transformation is divided into two steps. During the first step the UML diagrams are automatically translated using the ArgoSPE plugin [14]. During the second step, OCL constraints are manually transformed following the rules described in [1], which are based on the guidelines given in [15]. The results of the two steps are then merged using the algebra tool of GreatSPN [16]. 4) The *verification & validation* task is aimed at verifying via GreatSPN tool that the safety requirements are satisfied. In the case that the design does not meet the safety requirements, systematized recommendations to improve the design are formulated and a new iteration is carried out.

MDSafeCer - MDSafeCer (Model-driven Safety Certification) [3] is a method that adopts MDE principles to enable the semi-automatic generation of composable process-based argument-fragments within safety cases. Via MDSafeCer, process models compliant with a process modeling language meta-model (e.g., SPEM 2.0) are transformed into argumentation models compliant with SACM and presented via for instance GSN-goal structures. MDSafeCer generates process arguments based on a possible argumentation pattern, which is constituted of a top level claim stating that "the adopted *p* process is in compliance with the required {S} of standard- level {intLev}", where p, S, L are variables indicating respectively a specific process, a set of standards, a specific integrity level. This claim can be decomposed by showing that all the process activities have been executed and that in turn for each activity all the tasks have been executed and so on until an atomic process-related work-definition unit is reached.

3 Collecting and Modeling Elements of Compliance

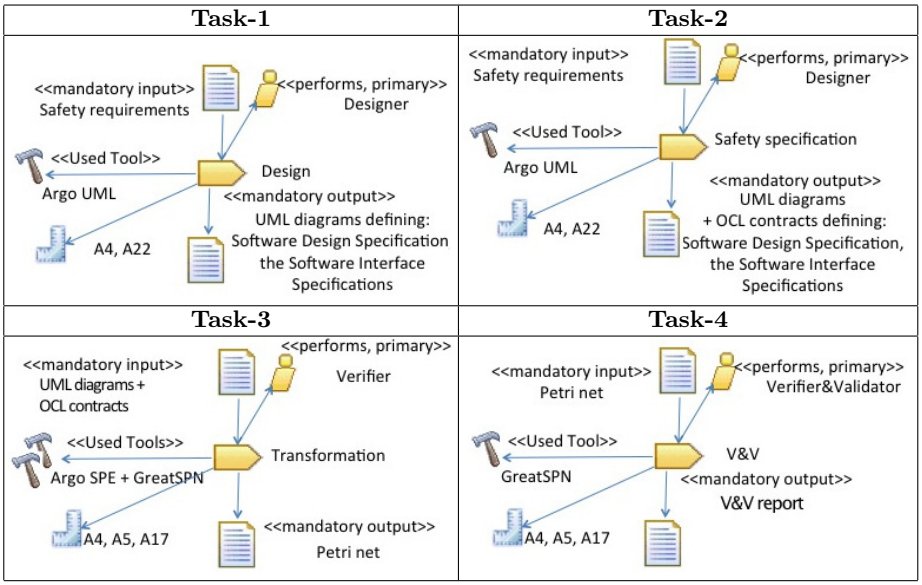
To partly act as safety and quality management evidence, needed for process assessment, MBASafe must be the result of the selection and composition of process elements that can be considered compliant with respect to the CENELEC series. MBASafe is a methodology to be used at design phase. Thus, first, it should be aligned with the Software Architecture & Design Phase. As recalled in Section 2, according to the CENELEC EN 50128, this phase should be carried out by appropriate roles, according to specific guidelines, be constituted of specific tasks, consume and produce specific work products. Since, as recalled in Section 2, MBASafe contains some of the required elements, its compliance can be partially argued about. More specifically, the following list highlights the process elements that meet the EN 50128 requirements: all the tasks that compose MBASafe can be considered aligned with he Software Architecture & Design Phase. However, not all the required tasks are included in MBASafe. This means that a company should be aware about what else should be performed. The task Transformation is not included in EN 50128 as a standalone task. It is implicitly expected to be executed (manually or automatically) in the

case of usage of formal methods within the verification task. Also the current sets of MBASafe in/out work products can be aligned. However, the EN 50128 expected number of in/out work products is greater. MBASafe guidelines can be aligned. As seen in the background formal methods and more specifically petri nets are among the techniques suggested to perform verification. With respect to roles, MBASafe does not pose enough emphasis. Nothing about qualifications is defined. Finally, the current tools (e.g., translator, model checker, etc.) that are proposed to perform the tasks do not offer satisfying evidence concerning their quality. Thus, MBASafe as it is cannot be adopted in real settings.

To enable its usage in real settings, the presentation of MBASafe should be enhanced and its alignment clearly made explicit. More specifically, all input/output work products should be specified and aligned with EN 50128. Concerning roles, vagueness in terms of their responsibility and degree of independence should be eliminated. Concerning tools, rational and adequate justifications in terms of their quality should be provided. In alternative, other tools should be suggested. In Table 2, we illustrate the SPEM 2.0 models representing the augmented MBASafe tasks.

By construction, these augmented MBASafe tasks contain process elements that are in compliance with EN 50128. To explain this compliance, in Section 4 we derive process-based arguments and we document them in GSN. Besides the enhancement of the presentation, to satisfy all the EN 50128, MBASafe should, however, be further developed or combined with another methodology offering complementary support. Thus, given the awareness developed thanks to the performed gap analysis, we also indicate the undeveloped goals.

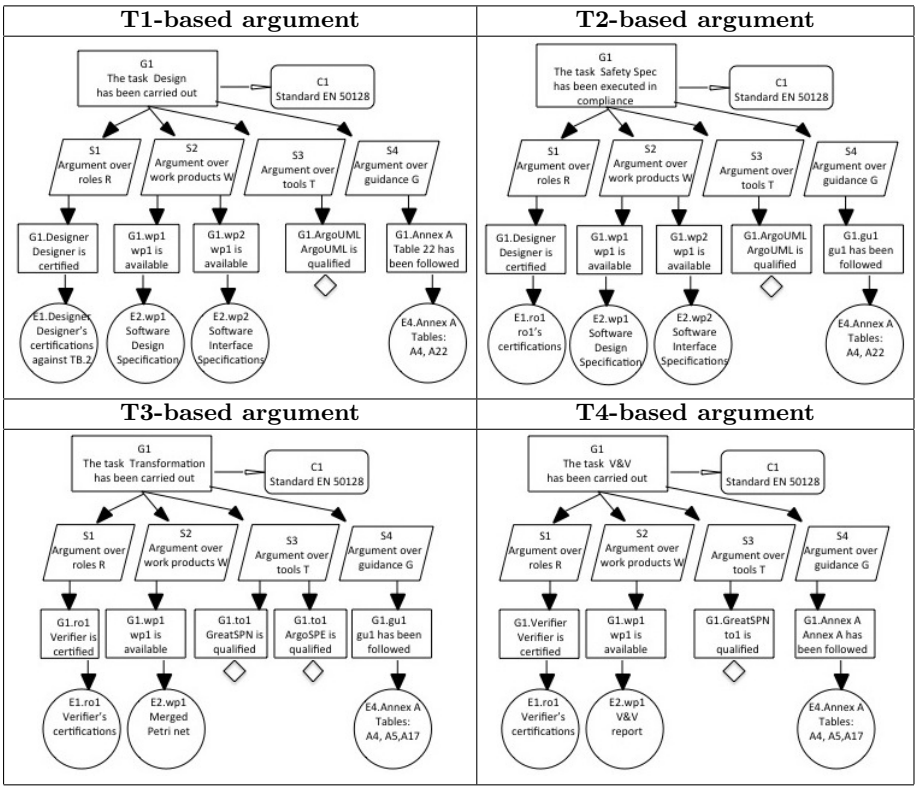
Table 2. MBASafe tasks given in SPEM 2.0



4 Arguing about EN 50128 Compliance via MDSafeCer

The aim of this section is to derive a process-based argument for arguing about MBASafe compliance with EN 50128. More specifically, our derived argument given in GSN argues that MBASafe is partially compliant with the EN 50128 requirements related to the design phase for a SIL2 subsystem. To derive such argument, we proceed compositionally and from the process models given in Table 2, by using MDSafeCer, we first derive sub-arguments that argue about compliance at task level. The derived sub-arguments are depicted in Table 3. Such arguments could be further developed to indicate the missing evidence (e.g., the missing work products).

Table 3. Task-based arguments



To argue at phase level, the rules that were initially proposed by Gallina [3] need to be further developed. More specifically, we present additional rules that are needed to generate a pattern instance based on our pattern on *Process compliance*, represented in Fig. 2 and in Fig. 3, whose structure partially borrows from the the *Goal decomposition* pattern and incorporates the divide and con-

quer principle. For sake of clarity, it should be stated here that the semantic mapping was previously given and explained [3].

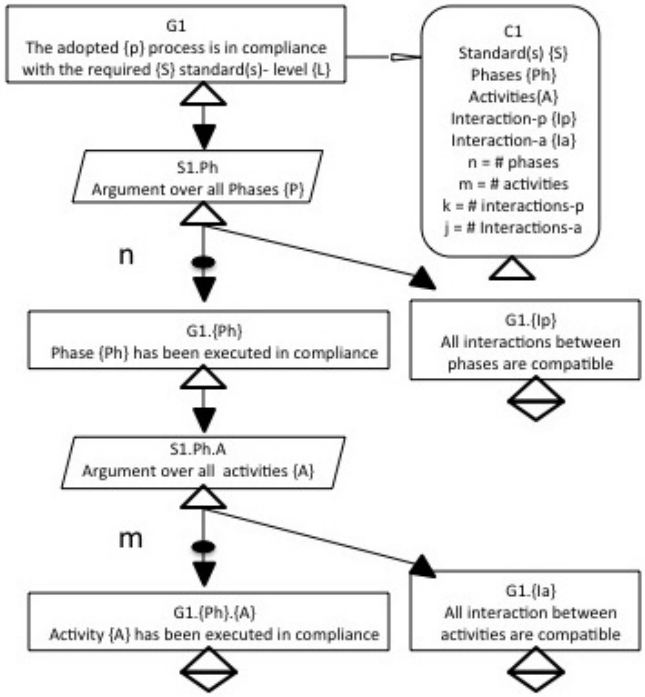


Fig. 2. Goal structure representing the *Process compliance* argumentation pattern.

For space reasons, Fig. 2 represents a pattern that considers only a 3-layer work-breaking-down structure. A process is divided into phases, which in turn are divided into activities. A richer hierarchy could be considered by breaking activities down further into tasks and finally tasks into steps. The 3-layer granularity is however sufficient for this paper since MBASafe can be considered a 2-layer hierarchy, i.e., a phase constituted of four activities. The four activities are named tasks in accordance with SPEM 2.0 models.

The additional needed rules are:

1. Create the top-level goal ID:G1 and statement: "The adopted p process is in compliance with the required {S} standard- level {intLev}". Create the context to be associated to G1. Context ID:C1 and statement: "Standard {S}", where S and L are variables. Create an inContextOf link to relate G1 and C1.

Develop the goal G1 further by creating one strategy.

(a) S1: "Argument over phases P".

2. Further develop strategy S1 by creating:

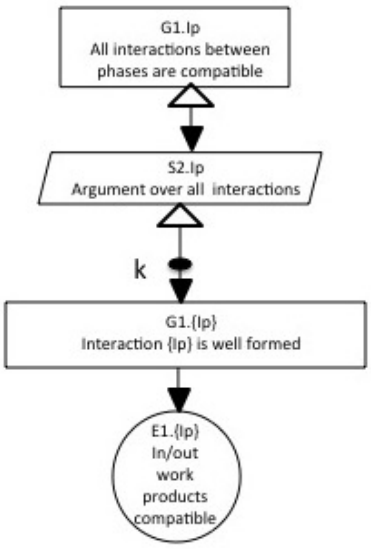


Fig. 3. Goal structure, continuation of the pattern in Fig. 2.

(a) for every phase *ph* in *P*, a goal G1.ph "Phase *ph* has been executed in compliance" and develop this goal further by creating an equivalent structure related to the lower level work decomposition".

(b) a goal G1.ip "All interactions {Ip} between Phases are compatible" and develop this goal further by creating one strategy: S2.Ip: "Argument over all interactions {Ip}". Further develop strategy S2.Ip by creating for every existing relation (representing an interaction between two phases) a goal G1.Ip "Interaction Ip is well formed" and develop this goal further by creating the corresponding solution E1.Ip "*Ip* In/out work products compatible" and the supportedBy link necessary to link S1.Ip with E1.Ip.

By aligning MBASafe-hierarchy with the pattern hierarchy and by manually following the above listed rules, we can easily derive the argument at the phase level, depicted in Fig. 4 and in Fig. 5 (note that for space reasons Fig. 5 does not present all the developed goals related to all the relations among tasks).

This argument can be easily composed with the sub-arguments, which were illustrated in Table 3. The compositional nature could be presented in a more advanced way by using modularized goal structures. Similarly, contracts could be used to clearly state the assumptions and guarantees that may exist between two sub goal-structures. In the context of distributed and heterogeneous management, where the responsibility for the provision of the different justifications might also be distributed and then integrated, contract-based goal structuring could be a winning solution.

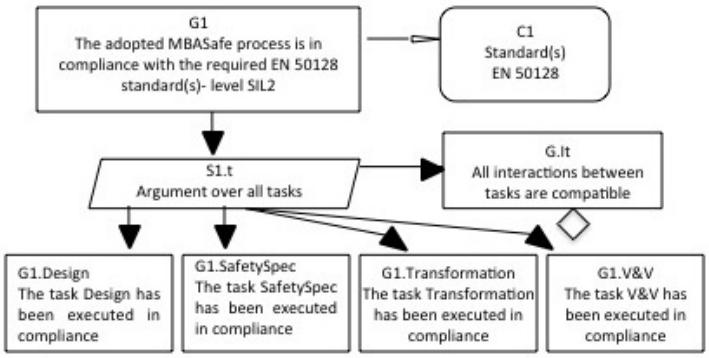


Fig. 4. Goal structure representing the argumentation pattern instance.

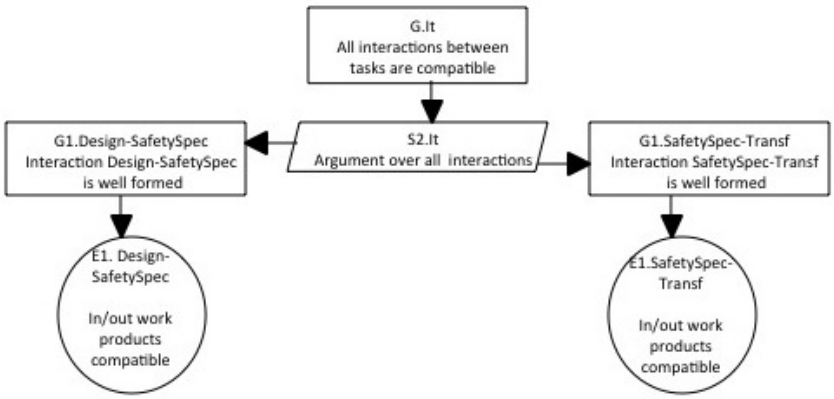


Fig. 5. Goal structure, continuation of the argumentation pattern instance.

5 Related Work

The current certification framework is traversing a crisis phase due to the growing complexity associated to the safety justifications that are required by the standards [17]. A balance between process and product-based justification is still not clear. Despite its necessity, process-based justification is proportionally less investigated. Bender et al [18] in their work on the certification nature, conclude that for the time being process adherence (including personal qualifications), classified as indirect evidence, must be provided. They however do not propose any process-related argument. More recently, Nair et al. [19] recognize the relevance of process-based argumentation and similarly to what proposed by Gallina [3] argue about the core process elements. Nair et al. call the process-based argument as secondary confidence argument. In its effort aimed at strengthening via process-based evidence an existing method that targets provision of product-based evidence, our work represents a novelty and an effort to contribute to the

achievement of the right balance. The possibility, in a long term, of deriving semi-automatically process-based arguments related to MBASafe will free time to be dedicated to the provision of product-based arguments.

6 Conclusion and Future Work

Since newly proposed and potentially innovative engineering methods suffer of low acceptance in rail industrial settings due to the requirements of the certification process, methods aimed at speeding up the provision of process-based arguments can be beneficial. In this paper, we have used MDSafeCer to show that MBASafe can be partly used as quality and safety management evidence within a safety case. More specifically, we have focused on specific portions of the CENELEC standard series related to software process compliance and we have argued by using GSN about compliance with EN 50128-related design.

In the future, to achieve a full compelling process argument, we will further develop MBASafe according to the findings. Ideally, all undeveloped goals should be replaced by well-founded and explained goals. Moreover, with respect to tool-support, in the context of SafeCer [20], a prototype implementation of MDSafeCer was integrated within Workflow Engine for Analysis, Certification and Testing (WEFACT), which is a tool that offers a flexible infrastructure for defining and executing processes as well as integrating other tools for rendering purposes. This implementation is expected to evolve in the framework of the recently funded ECSEL project AMASS. The initial goal of its evolution is to provide evidence with respect to the effectiveness of the approach in terms of time reduction (manual vs. semi-automatic work). Once the evidence is achieved, the intention is to provide an industry-friendly tool support. As future work, we also aim at focusing on evidence related to the system/subsystem behavior, i.e., technical evidence. To do that, we plan to derive product-based arguments by building on top of work presented by Sljivo et al. [21].

Acknowledgments. This work has been partially supported by the ARTEMIS project nSafeCer [20] and by the Swedish Foundation for Strategic Research via the SYNOPSIS project [22] and the Gen&ReuseSafetyCases project [23].

References

1. Gómez-Martínez, E., Rodríguez, R.J., Etxeberria Elorza, L., Illarramendi Rezabal, M., Benac Earle, C.: Model-based verification of safety contracts. In Canal, C., Idani, A., eds.: Software Engineering and Formal Methods. Volume 8938 of LNCS. Springer (2015) 101–115
2. Object Management Group: Software & Systems Process Engineering Meta-Model (SPEM), v2.0. Full Specification formal/08-04-01. (2008)
3. Gallina, B.: A model-driven safety certification method for process compliance. In: 2nd Int. Workshop on Assurance Cases for Software-intensive Systems (ASSURE). (Nov. 2014) 204–209

4. Gallina, B., Provenzano, L.: Deriving Reusable Process-based Arguments from Process Models in the Context of Railway Safety Standards. AUJ **36**(4) (2015)
5. : Interim Defence Standard 00-56 Part 1 - Issue 5, in, UK MOD (2014)
6. GSN: Community Standard Version 1 (2011)
7. SACM: http://www.omg.org/spec/sacm/1.0
8. MODSafe Modular Urban Transport Safety and Security Analysis: Survey of current safety lifecycle approaches, DEL D6.1 TRIT WP6 100531 V1.0. Technical report (2010)
9. BS EN50126: Railway applications: The specification and demonstration of Reliability. Availability, Maintainability and Safety (RAMS) (1999)
10. BS EN50129: Railway applications Communication, signalling and processing systems Safety related electronic systems for signalling (2003)
11. BS EN50128: Railway applications - Communication, signalling and processing systems Software for railway control and protection systems (2011)
12. Söderberg, A., Johansson, R.: Safety Contract Based Design of Software Components. In: IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW). (2013) 365–370
13. Ajmone Marsan, M., Balbo, G., Conte, G., Donatelli, S., Franceschinis, G.: Modelling with Generalized Stochastic Petri Nets. Wiley Series in Parallel Computing. John Wiley and Sons (1995)
14. Gómez-Martínez, E., Merseguer, J.: ArgoSPE: Model-based Software Performance Engineering. In: Proc. 27th Int. Conf. on Applications and Theory of Petri Nets and Other Models of Concurrency (ICATPN). Volume 4024 of LNCS., Springer (2006) 401–410
15. Liu, T.S., Chiou, S.B.: The application of Petri nets to failure analysis. Reliab. Eng. Syst. Safe. **57**(2) (1997) 129–142
16. Baarir, S., Beccuti, M., Cerotti, D., De Pierro, M., Donatelli, S., Franceschinis, G.: The GreatSPN tool: recent enhancements. SIGMETRICS Perform. Eval. Rev. **36**(4) (2009) 4–9
17. Gallina, B.: How to increase efficiency with the certification of process compliance. In the 3rd Scandinavian Conference on SYSTEM & SOFTWARE SAFETY, Stockholm, March 24-25 (2015)
18. Bender, M., Maibaum, T., Lawford, M., Wassyng, A.: Positioning verification in the context of software/system certification. In: 11th International Workshop on Automated Verification of Critical Systems (AVOCS), Newcastle upon Tyne (UK), Sept. 12-15. (2013)
19. Nair, S., Walkinshaw, N., Kelly, T., de la Vara, J.L.: An evidential reasoning approach for assessing confidence in safety evidence. In: IEEE 26th International Symposium on Software Reliability Engineering (ISSRE). (Nov 2015) 541–552
20. ARTEMIS-JU-269265: SafeCer-Safety Certification of Software-Intensive Systems with Reusable Components. http://www.safecer.eu/
21. Sljivo, I., Gallina, B., Carlson, J., Hansson, H.: Generation of Safety Case Argument-Fragments from Safety Contracts. In: 33rd International Conference on Computer Safety, Reliability, and Security. Volume 8666 of Lecture Notes in Computer Science., Springer (September 2014) 170–185
22. SYNOPSIS-SSF-RIT10-0070: SYNOPSIS project-safety Analysis for Predictable Software Intensive Systems. Swedish Foundation for Strategic Research
23. Gen&ReuseSafetyCases-SSF: http://www.es.mdh.se/projects/393-genreusesafetycases