Reliability-Aware Design Specification for Allowing Reuse-Based Reliability Level Increment

Work in progress

Patricia López
Tekniker
Eibar, Spain
e-mail:patricia.lopez@tekniker.es

Leire Etxeberria, Xabier Elkorobarrutia
Electronics and Computing Department
Mondragon Unibertsitatea, Engineering Faculty
Mondragon, Spain
e-mail:{letxeberria,xelkorobarrutia}@mondragon.edu

Abstract— **The development of safety-critical systems is expensive and reuse can be seen as a way of reducing the development cost of safety-critical systems. In this context, models could be helpful for safety-critical system development and also to facilitate safe reuse. In this paper, an approach for allowing the reuse-based reliability level increment is presented. This approach is based on a holistic reliability-aware design specification which is related to reliability levels using a knowledge base.**

Keywords-Reliability; safety; reuse; model-based .

I. INTRODUCTION

Cyber-Physical Systems (CPS) are embedded ICT systems that are interconnected, interdependent, collaborative, and autonomous. They provide computing and communication, monitoring/control of physical components/processes in various applications including safety critical. Safety is a key aspect of Safety-critical CPSs. A safety-critical CPS is a CPS whose failure or malfunction may result in death or serious injury to people, loss or severe damage to equipment/property or environmental harm.

The cost of developing safety-critical CPSs is much higher than the cost of developing other kind of software. “A commonly accepted rule of thumb is that development of safety-certified software costs roughly 10 times, as much as non-certified software with equivalent functionality” [1]. Moreover, CPSs have usually real-time constraints and this increases the complexity, “the cost of developing safety-critical software is likely to be 20 to 30 times the cost of developing typical management information software” [1].

Evolution of products is also more costly in safety-critical systems as the re-certification may imply very time-consuming re-doing activities such as re-design, re-verification and re-validation.

Reuse can be seen as a way of reducing development (and specially re-development) costs of safety-critical systems. However, reuse is quite challenging in safety-critical domains as safety must be guaranteed.

Safety-critical systems are developed following domain-specific safety standards that rule what kind of techniques must be used depending on the reliability level to be obtained and safety argumentation is made based on a specific context. And reuse implies to change the context or reliability level.

Models could be helpful for safety-critical system development and also for facilitate reuse. Model-Driven Engineering (MDE) refers to the systematic use of models as primary engineering artifacts throughout the engineering lifecycle. The complexity of system engineering is increasing and model-driven engineering helps to deal with this increasing complexity. For the development of safety-critical systems, MDE could be used for different purposes [2][3]:

- MDE-based development of safety-critical systems: MDE used during the development process of systems for development, verification and validation purposes.
- MDE-based safety certification: MDE for managing safety evidences, MDE for supporting the verification of compliance to safety standards, etc.

This paper presents a model based approach for supporting the reuse of safety critical systems with a special focus on facilitating the increment of reliability level when a product is reused.

Section II presents the state of the art in the area, section III presents the Model-based Approach for Reuse-based Reliability level Increment, section IV addresses the case study that has been used and to finish the conclusions and future work section.

II. STATE OF THE ART

A. Reuse in safety-critical systems

Reuse in safety-critical systems is a research topic that has received quite attention lately. European projects, such as Safety Certification of Software-Intensive Systems with Reusable Components (SafeCer) or Open Platform for EvolutioNary Certification Of Safety-critical Systems (Opencoss) have been focused on reusability of safety critical systems.

There are different reuse scenarios in safety: Intra-standards when reuse is done in the same domain and to meet the same standard or inter-standard or cross domain when a component or system is reused in another domain and must meet another standard.

In the intra-standard scenario, the reason of reusing could be also different: evolutionary scenario when a system or component changes and we need to assure that is safe, a new product with slightly different requirements, a family of products, when the standard evolve (new version of the standard), when we want to increment the reliability level, etc.

Different kinds of artifacts could be reused as well: requirements [4], components [5], system, safety argumentation, safety case [6][7][29], hazard analysis [8][9]… Depending on what is reused, the phase of the life cycle where is reused is also different; mainly two broad phases could be distinguished: Reuse during construction of the system according to the safety requirements or Reuse during accreditation and certification of the system: providing evidence.

B. Reliability levels

“Traditionally, certification standards have been process-oriented, i.e., where a hazard analysis is performed to identify the severity and risks associated in functional failure for determining a Safety Level, which in turn is used to choose and customize the process applied” [10]. This safety level specifies a target level of risk reduction.

These safety or reliability levels are different depending on the domain-specific standard. IEC 61508 standard, who is intended to be a basic functional safety standard applicable to all kinds of industry, defines the Safety integrity levels (SIL). There are four discrete integrity levels associated with SIL with SIL 4 the most dependable and SIL 1 the least. The SIL can be assigned to any safety relevant function or system or sub-system or component. The

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. In the standard each SIL level is associated to a set of measures to be implemented into the design during the design process.

The standards derived from IEC 61508 such as the standards for industrial processes (IEC 61511), or railway industry (EN 50126/EN 50128 /EN 50129) also use SIL.

Other standards specified other levels. In the automotive domain (ISO 26262), the Automotive Safety Integrity Level (ASIL) is used, a risk classification scheme that is an adaptation of the SIL for the automotive industry. The ASIL is established by performing a risk analysis of a potential hazard by looking at the Severity, Exposure and Controllability of the vehicle operating scenario. The safety goal for that hazard in turn carries the ASIL requirements. There are four ASILs identified by the standard: ASIL A, ASIL B, ASIL C, ASIL D. ASIL D dictates the highest integrity requirements on the product and ASIL A the lowest.

For airborne systems (the DO-178C and DO-254 standards) Design Assurance Levels (DAL) are proposed. The DAL is determined from the safety assessment process and hazard analysis by examining the effects of a failure condition in the system. There are five levels of compliance, A through E, which depend on the effect a failure will have on the operation of the aircraft. Level A is the most stringent, defined as "catastrophic" (e.g., loss of the aircraft), while a failure of Level E will not affect the safety of the aircraft.

The different kind of levels could be compared as they have some similarities, but they have also differences; there is not a one-to-one mapping.

Apart from standards, at OPENCOSS project they have developed the concept of Assured reliability and Resilience Level (ARRL) of components [11]. It is an approach that is not applied at system level but at component level, which helps to compose safe systems from components. It is based on the Quality of Service of a component, which is a more generic criterion that takes the trustworthiness as perceived by users better into account. This concept complements the Safety Integrity Level concept.

C. Reliability or Certification-aware design specification

As stated in [12] “*Unfortunately, little work has been done to date on accommodating the additional demands that certification imposes on how the design of systems should be expressed. Our experience indicates that certification is often (incorrectly) viewed as an after-the-fact activity. This can give rise to various problems during certification, because a large fraction of the safety evidence necessary for certification has to be gathered during the design phase and embodied in the design specification. Failing to make the design “certification-aware” will inevitably lead to major omissions and effectively make the design “unauditable” for certification purposes.*”

In [12], they propose a methodology and guidelines for modeling Software-Hardware Interfaces using SysML (Block Definition Diagrams, Internal Block Diagrams, Activity Diagrams and Requirement Diagrams). The goal is to describe the design and establish the traceability (link requirements and design).

Although [12] introduced the concept of “certification-aware design specification” and proposed a methodology, not all the aspects needed to get a reliability-aware design specification are covered. To the best of our knowledge, there is not a holistic approach for specifying a reliability-aware specification.

This design specification should be Product-aware and also Process-aware. Product-aware specification should provide aspects, such as requirements-design traceability, test case-requirements traceability, the applied fault tolerance techniques reflected in the design, failure modes linked to design elements, properties and contracts (formal methods) linked to design elements and requirements…

The process-aware specification should include information about the safety standards that have been applied, the reliability level, the used techniques in the phases of the life cycle and the link to the results of the applied techniques (some aspects specified in the product-aware part, testing results, results of formal proofs…).

There are approaches that cover part of the needs of a holistic reliability-aware design specification:

For requirement analysis and modeling requirement traceability [12][13][14][15][16], etc. For adding formal properties or contracts to the specification: [17][18]. There are a lot of approaches for relating safety analysis concepts and design specification or transforming the design in safety analysis concepts: [19][20][21][22], etc. For specifying fault tolerance techniques, safety patterns could be used [23] presents an approach for representing Safety Patterns in a design. Regarding Process aware specification, [24] presents a domain model of IEC 61508 concepts: Domain model for SIL activities, Domain model for certification, Domain model for communication, etc. And [25][26] present a conceptual model of evidences for safety cases.

III. Model-based Approach for Reuse-based Reliability level Increment

Our approach is based on the following hypothesis:

- To provide a “reliability-aware” design specification helps to reason about the reliability level of a system or component. This can facilitate certification process, the reuse of components and reliability increasing process.
- It is possible to define reliability levels of components and systems and relate this reliability levels to techniques applied during design. Therefore, it is possible to define a decision system that helps to decide which techniques to apply to increase reliability.

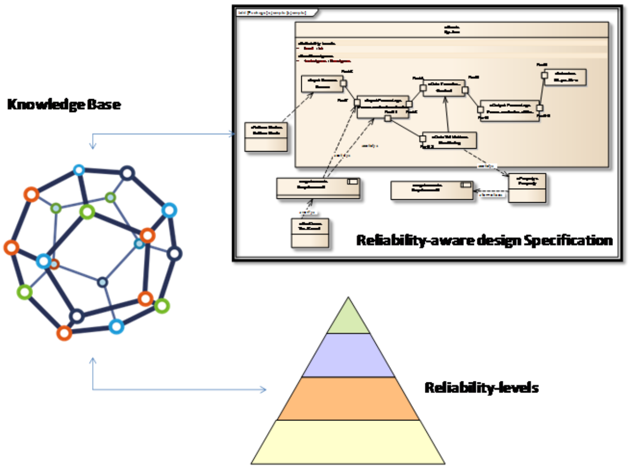

Figure 1. Architecture of the approach

The approach proposes to use a reliability-aware design specification in combination with a reliability-level classification and a knowledge base that relates the levels and the techniques applied and modeled in the specification (see Figure 1).

The main goal of the approach is to facilitate the increment of the reliability level of a system. In industry often it is required to develop a new system with same functionalities as a previous one but with a higher reliability level. The approach will help to reuse the design, verification, validation and certification artifacts of the existing system to a point avoiding expensive re-design and re-certification activities from scratch.

A. *Reliability-aware design specification*

The proposed specification is based on SysML and existing approaches has been reviewed, selected and combined to support a holistic reliability-aware view. System Modeling Language (SysML) is a graphical modeling language for System Engineering. It can be considered as an extension of UML2 for systems. It supports the specification, analysis, design, verification, and validation of systems that include hardware, software, data, personnel, procedures, and facilities. SysML is a Critical Enabler for Model Driven System Engineering. SysML could be considered the de-facto standard for systems engineering [27]. Moreover, SysML is rapidly becoming the notation of choice for developing safety-critical systems [13].

The specification has two differentiated parts: the Product-aware specification and the Process-aware specification.

For the Product-aware specification, the following aspects are modeled:

Structural modeling is done using Block Definition Diagrams (bdd) and Internal block diagrams (ibd) of SysML. SysML employs the concept of blocks to specify hierarchies and interconnection within a system design. A BDD describes the system hierarchy and system/component classifications; it lets you describe relationships between blocks, such as composition, association, and specialization. Whereas the IBD describes the internal structure of a system in terms of its parts, ports, and connectors. Interfaces are described using the Port concept of ibds.

For *requirements*, the SysML Requirements diagram is used. Requirements diagram is an extension of the class diagram that allows the modelling of detailed system requirements. It represents the system requirements and their relationships. *Traceability* links are gathered in the diagram: among requirements, among requirements and test cases, among requirements and design and among requirements and other model elements (use cases…). *Test cases* are modeled as special blocks with <<Test Case>> stereotypes to allow the traceability to requirements and design blocks.

Formal properties proven using formal methods are also modeled using an adaptation of the proposal of [17]. Properties are traced to design elements and requirement blocks.

Fault tolerance techniques such as monitors or replication are modeled using safety patterns [23].

And design elements are trace to *failure modes.*

The Process-aware specification includes:

- The *reliability level* assigned to the component or system
- The *standard* applied
- The *list of techniques* applied in each phase
- And *links to the product-aware part* and *results* (testing results, results of formal proofs…).

Meta-data in the Sysml model is used for specifying process-aware information for example using attributes with stereotypes in a block (see figure 2).

B. *Reliability levels*

Reliability levels will be defined for components and systems. This will be done based on ARRL [11] as it provides the reliability level at component level and SIL levels.

C. *Decision system for increasing reliability*

A decision system is being developed that will support reuse, especially increasing reliability level of a component/system.

Based on the reliabilility-aware design specification is possible to know the applied techniques and results and assign a reliability level.

A knowledge base will be developed for being able to relate reliability levels and techniques and guide the increment of reliability. This base will help to answer the following questions:

- Which techniques should be applied to increase reliability?
- Which is the current level of reliability of a design?
- …

IV. CASE STUDY

The approach is being applied to a case study. As first case study, an educational use case has been selected [28]. This educational demonstrator has been previously used in lectures related to safety, real-time, software engineering and embedded system development. It is based on an elevator system control. The elevator system is composed of 2 or more elevators and they lift or bring down a load in a coordinated way. Each elevator has attached a motor, up and down sensor and shaft rotation sensor that is used to infer position and speed.

Each elevator is controlled by an ElevatorCtrl software component. It reads from its sensor, actuates on its motor and announces its state to the main controller. All elevator coordination is in charge of ElevatorSystemCtrl. The one that commands all the elevators on response to an operator. The operator has an interface for commanding the system.

The system is assigned next safety requirements:

- If one crane/elevator stops, the others must stop within 50 millisecond.
- The difference of position between two elevators can't be greater than 10 mm.

Depending on the context where this system will be used, the required reliability level could vary. A first version of the design has been specified using the reliability-aware design specification.

This specification gathers the design of the system (components, interfaces, ports, etc.) using SysML. The traceability information has been also captured: requirements traced to other requirements (some requirements are derived from the "If one elevator stops, the others must stop within 50 milliseconds" requirements), requirements traced to the test cases that verify the requirement and requirements trace to the design elements (components, ports…) that satisfy the requirement. Formal properties such as safety contracts that have been used for verification of timing have been also specified.

Safety patterns applied to the design are showed explicitly such as the Monitor pattern of the Communication Supervisor used in the system.

Finally, metadata is used to add information about the process such as the required reliability level and the used techniques.

The figure 2 is an excerpt of the reliability-aware design specification (concepts that were captured in different diagrams have been mixed for presentation purposes).

One of the benefits of having a reliability-aware design specification is that it facilitates the reuse of the system's design, the reuse of verification & validation artifacts and also the reuse of certification artifacts.

As next step, a scenario of reusing with an increment of reliability level is foreseen.

V. CONCLUSIONS AND FUTURE WORK

First results, especially of the reliability-aware design specification show interesting findings. The approach could be useful for reusing the design with different purposes not only for incrementing reliability. Moreover, the approach is also useful for novel safety engineers or companies that start developing safety-critical systems but they have not so much experience with standards.

However, we have only preliminary results with an educational case study. Further work is needed to see the applicability of the approach in industry.

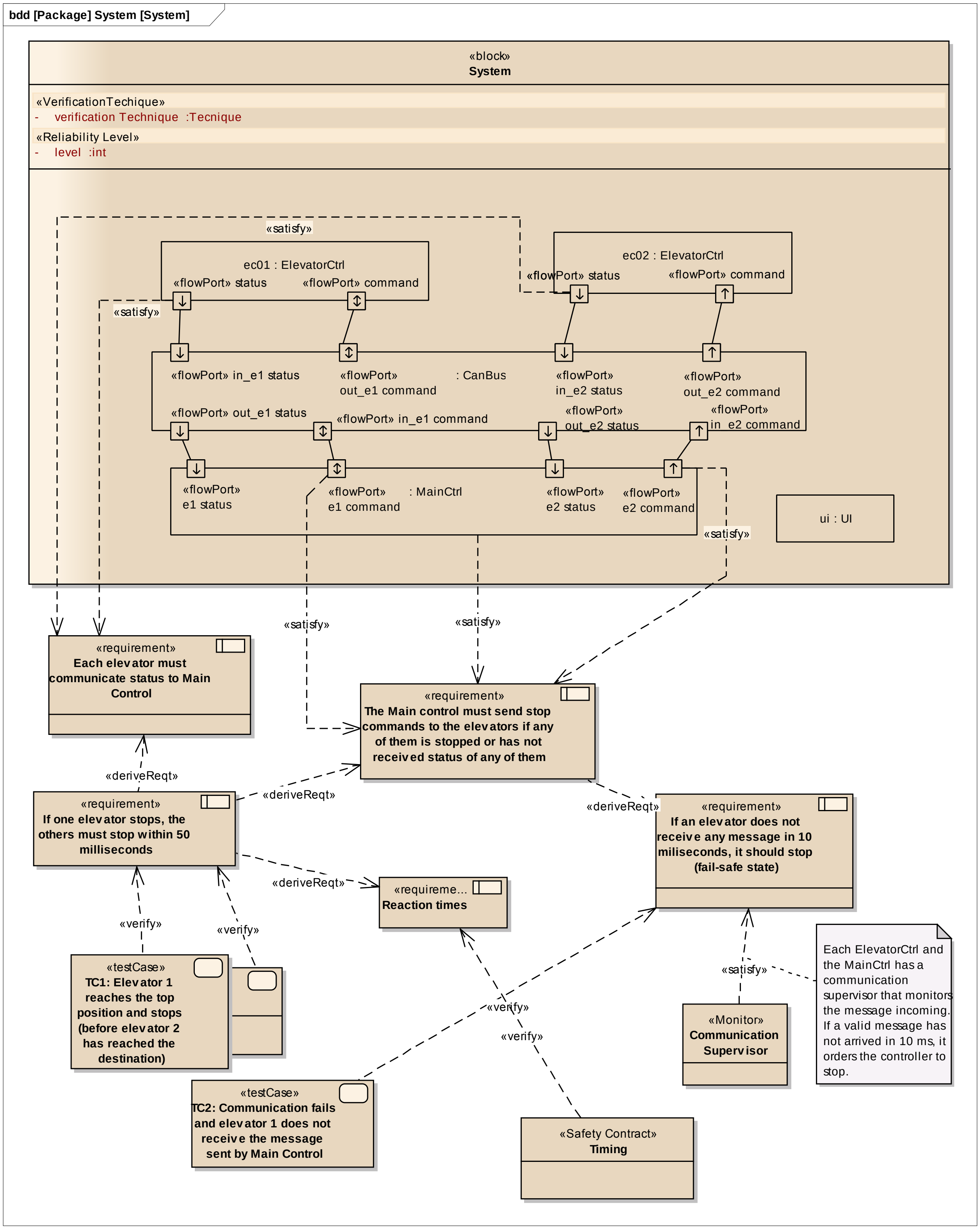
bdd [Package] System [System]
«block»
System
«VerificationTechique»
- verification Technique :Tecnique
«Reliability Level»
- level :int
«satisfy»
ec01 : ElevatorCtrl
«flowPort» status
«flowPort» command
«satisfy»
ec02 : ElevatorCtrl
«flowPort» status
«flowPort» command
«flowPort» in_e1 status
«flowPort» out_e1 command
: CanBus
«flowPort» in_e2 status
«flowPort» out_e2 command
«flowPort» out_e1 status
«flowPort» in_e1 command
«flowPort» out_e2 status
«flowPort» in_e2 command
«flowPort» e1 status
«flowPort» e1 command
: MainCtrl
«flowPort» e2 status
«flowPort» e2 command
«satisfy»
ui : UI
«satisfy»
«satisfy»
«requirement»
Each elevator must communicate status to Main Control
«requirement»
The Main control must send stop commands to the elevators if any of them is stopped or has not received status of any of them
«deriveReqt»
«deriveReqt»
«deriveReqt»
«requirement»
If one elevator stops, the others must stop within 50 milliseconds
«requirement»
If an elevator does not receive any message in 10 miliseconds, it should stop (fail-safe state)
«deriveReqt»
«requireme...
Reaction times
«verify»
«verify»
«verify»
«verify»
«satisfy»
«testCase»
TC1: Elevator 1 reaches the top position and stops (before elevator 2 has reached the destination)
«Monitor»
Communication Supervisor
Each ElevatorCtrl and the MainCtrl has a communication supervisor that monitors the message incoming. If a valid message has not arrived in 10 ms, it orders the controller to stop.
«testCase»
TC2: Communication fails and elevator 1 does not receive the message sent by Main Control
«Safety Contract»
Timing

Figure 2. Excerpt of the reliability-aware design specification of the case study

REFERENCES

[1] K. Nilsen, Certification Requirements for Safety-Critical Software, RTC magazine, 2004, http://www.rtcmagazine.com/articles/view/100010, retrieved: October, 2015.

[2] J. L. de la Vara et al, Towards a model-based evolutionary chain of evidence for compliance with safety standards. In "Proceedings of the 2012 international conference on Computer Safety, Reliability, and Security (SAFECOMP'12)", F. Ortmeier and P. Daniel (Eds.). Springer-Verlag, Berlin, Heidelberg, 2012, pp.64-78.

[3] R. K. Panesar-Walawege, Using model-driven engineering to support the certification of safety-critical systems, Doctoral thesis, University of Oslo, 2012

[4] J. Dehlinger and R.R. Lutz. 2005. A product-line requirements approach to safe reuse in multi-agent systems. "SIGSOFT Softw. Eng. Notes" 30, 4, 2005,pp. 1-7.

[5] R. Land, M. Åkerholm, and J. Carlson. 2012. Efficient software component reuse in safety-critical systems - an empirical study. In "Proceedings of the 31st international conference on Computer Safety, Reliability, and Security (SAFECOMP'12)", F. Ortmeier and P. Daniel (Eds.). Springer-Verlag, Berlin, Heidelberg, 2012, pp. 388-399.

[6] P. Fenelon, T. P. Kelly, and J. A. McDermid, Safety Cases for Software Application Reuse. In the "proceedings of SAFECOMP '95", 1995, pp. 419-436

[7] D. Bush, Towards Formalising Reuse in Safety Cases, "Proceedings of the INCOSE UK Spring Symposium", Tolleshunt Knights, Essex, 2002.

[8] S. Baumgart, Investigations on hazard analysis techniques for safety critical product lines, "IDT Workshop on Interesting Results in Computer Science and Engineering (IRCSE)", 2012.

[9] L. Grunske, B. Kaiser, and R. H. Reussner, Specification and evaluation of safety properties in a component-based software engineering process. In "Component-Based Software Development for Embedded Systems", C. Atkinson, C. Bunse, H. G. Gross, and C. Peper (Eds.). Springer-Verlag, Berlin, Heidelberg, 2005, pp. 249-274.

[10] SafeCer project (Safety Certification of Software-Intensive Systems with Reusable Components), http://safecer.eu/, retrieved: October, 2015.

[11] E. Verhulst and B. H. C. Sputh, , ARRL: A criterion for compositional safety and systems engineering: A normative approach to specifying components, "Software Reliability Engineering Workshops (ISSREW)", 2013 IEEE International Symposium on , vol., no., 4-7 Nov. 2013, pp.37-44

[12] M. Sabetzadeh, S. Nejati, L. Briand, and A. H. Evensen Mills, Using SysML for Modeling of Safety-Critical Software-Hardware Interfaces: Guidelines and Industry Experience. In Proceedings of the 2011 IEEE 13th International Symposium on High-Assurance Systems Engineering (HASE '11). IEEE Computer Society, Washington, DC, USA, 2011, pp.193-201.

[13] S. Nejati, M. Sabetzadeh, D. Falessi, L. Briand, and T. Coq. 2012. A SysML-based approach to traceability management and design slicing in support of safety certification: Framework, tool support, and case studies." Inf. Softw. Technol." 54, 6, 2012, pp. 569-590.

[14] D. Falessi, S. Nejati, M. Sabetzadeh, L. Briand, and A. Messina, SafeSlice: a model slicing and design safety inspection tool for SysML. In "Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering (ESEC/FSE '11)". ACM, New York, NY, USA, 2011, pp. 460-463.

[15] A. Albinet, J.-L. Boulanger, H. Dubois, M.-A. Peraldi-Frati, Y. Sorel, and Q.-D. Van, Model-based methodology for requirements traceability in embedded systems, in "Proceedings of 3rd European Conference on Model Driven Architecture Foundations and Applications, ECMDA'07", Haifa, Israel, 2007.

[16] P. Colombo, F. Khendek, and L. Lavazza, Requirements analysis and modeling with problem frames and SysML: a case study. In "Proceedings of the 6th European conference on Modelling Foundations and Applications (ECMFA'10)", T. Kühne, B. Selic, M.-P. Gervais, and F. Terrier (Eds.). Springer-Verlag, Berlin, Heidelberg, 2010, pp.74-89.

[17] J.-F. Pétin, D. Evrot, G. Morel, and P. Lamy, Combining SysML and formal models for safety requirements verification, "ICSSEA 2010", 2010.

[18] S. Tonetta, Contract-based design of safety-critical software components, "International Workshop on Critical Software Component Reusability and Certification across Domains (CSC 2013)", ICSR13 workshop, June 18 2013

[19] K. Thramboulidis and S. Scholz, Integrating the 3+1 SysML view model with safety engineering, "Emerging Technologies and Factory Automation (ETFA)", 2010 IEEE Conference on , vol., no., 1,8, 2010, pp.13-16

[20] G. Li and B. Wang, SysML aided safety analysis for safety-critical systems. In "Proceedings of the Third international conference on Artificial intelligence and computational intelligence - Volume Part I (AICI'11)", H. Deng, D. Miao, J. Lei, and F. L. Wang (Eds.), Vol. Part I. Springer-Verlag, Berlin, Heidelberg, 2011, pp. 270-275.

[21] F. Mhenni, N. Nguyen, H. Kadima, and J. Choley, Safety analysis integration in a SysML-based complex system design process, "Systems Conference (SysCon)", 2013 IEEE International, vol., no., 2013, pp.70-75

[22] J. Xiang, K. Yanoo, Y. Maeno, and K. Tadano, Automatic Synthesis of Static Fault Trees from System Models, "Secure Software Integration and Reliability Improvement (SSIRI)", 2011 Fifth International Conference on, 2011, pp.127-136

[23] P. Antonino, T. Keuler, E.Y. Nakagawa, , Towards an approach to represent safety patterns, "The Seventh International Conference on Software Engineering Advances, ICSEA", 2012.

[24] D. Kuschnerus, F. Bruns, T. Musch, A UML Profile for the Development of IEC 61508 Compliant Embedded Software, "Embedded Real Time Software and Systems - ERTS² ", 2012.

[25] R. K. Panesar-Walawege, M. Sabetzadeh, and L. Briand, Using UML Profiles for Sector-Specific Tailoring of Safety Evidence Information, "30th International Conference, ER 2011", Brussels, Belgium, October 31 - November 3, 2011, pp.362-378

[26] R. K. Panesar-Walawege, M. Sabetzadeh,L. Briand, T. Coq, Characterizing the Chain of Evidence for Software Safety Cases: A Conceptual Model Based on the IEC 61508 Standard, "Software Testing, Verification and Validation (ICST)", 2010 Third International Conference on , vol., no., 6-10 April 2010,pp.335-344

[27] W. Schafer, and H. Wehrheim, The Challenges of Building Advanced Mechatronic Systems, "Future of Software Engineering, FOSE '07", 23-25 May 2007, pp.72-84

[28] M. Illarramendi, L. Etxeberria, and X. Elkorobarrutia, Reuse in Safety Critical Systems: Educational Use Case First Experiences. In "Proceedings of the 2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA '14)". IEEE Computer Society, Washington, DC, USA, 2014, pp. 417-422.

[29] I. Sljivo, Facilitating Reuse of Safety Case Artefacts Using Safety Contracts, Doctoral thesis, Mälardalen University, 2015