COUPLING MODEL CHECKING AND PSA – A CASE STUDY

Kim Björkman
VTT Technical Research Centre of Finland Ltd., Finland. E-mail: kim.bjorkman@vtt.fi

Antti Pakonen
VTT Technical Research Centre of Finland Ltd., Finland. E-mail: antti.pakonen@vtt.fi

Performing exhaustive model checking for a digital instrumentation and control (I&C) system, when both hardware failures and the detailed functionality of the I&C system are considered, can be challenging due to scalability issues. In our previous work, we have proposed a coupling approach where the model checking analysis is restricted to a limited set of postulated hardware failures based on probabilistic safety assessment (PSA) results, potentially improving scalability of model checking. Based on the pilot case study using a small example system, the coupling approach should scale quite well to larger systems. In this paper, we study the applicability of the coupling approach by performing a case study using a more complex example system, based on a fictitious reference model of a boiling water nuclear reactor. Compared to unrestricted model checking analysis, the computation times decreased considerably when the analysis was restricted. Still, the overall process for introducing the failures requires a lot of manual labor, and needs to be automated.

Keywords: Model checking, PSA, digital I&C, hardware failure, verification, scalability.

1. Introduction

Model checking (Clarke et al. 1999) is a formal verification method. In model checking, a desired system property (hardware or software) is verified over a system model by exhaustively enumerating over of all reachable states and possible behaviors (Clarke et al. 1999). When the system model fails to satisfy a wanted property, the model checker (a software tool used for analysis) produces a counter-example demonstrating a scenario violating the property.

Exhaustive verification of digital instrumentation and control (I&C) systems can be challenging, especially when both hardware failures and detailed functionality of the I&C systems are taken into account. Already for a rather simple I&C design, the system model can become too complex. This can be an issue especially in safety critical domains, such as nuclear power.

In our previous work (Lahtinen and Björkman 2016b), we have proposed a concrete concept-level coupling approach, combining model checking and probabilistic safety assessment (PSA) for the verification of digital I&C systems. The main idea of the approach is to use PSA results to restrict the model checking analysis to a limited set of postulated component failures. Based on the small-scale case study, the approach seems quite promising.

In this paper, we study the applicability of the coupling approach by performing a case study using a more complex example system, based on a fictitious reference model of a boiling water nuclear reactor (Authén et al. 2015). The focus of the case study is on assessing the scalability of model checking when the analysis is restricted based on PSA results.

The rest of this paper is structured as follows. Section 2 discusses coupling between PSA and model checking. In section 3, the example system of the case study is summarized. The concept-level coupling approach is presented in section 4. In section 5, the case study is presented. Sections 6 and 7 conclude this paper.

2. Integrating Model Checking and PSA

2.1 *Related research*

We have reviewed related work in Björkman et al. (2015), and Lahtinen and Björkman (2016a, 2016b). Our aim is not to repeat the review but to extend it with some of the latest related research.

Large part of the current related research focuses on using model checking techniques for Model-Based Safety Analysis (MBSA) and computing minimal cut sets (MCS) (e.g. Bozzano et al. (2015a, 2015b)), or computing dynamic fault trees (e.g. (Volk et al. 2018)). In Chen et al. (2015), an approach for combining model checking and fault tree analysis for verifying software safety properties is presented. In the approach, the safety properties to be verified by model checking are generated from minimal cut sets computed from a sequence fault tree (fault tree included with features from computational tree logic of model checking).

2.2 *Coupling model checking and PSA*

There are, at least in theory, considerable similarities between model checking and PSA.

Proceedings of the 29th European Safety and Reliability Conference.
Edited by Michael Beer and Enrico Zio

Published by Research Publishing, Singapore.
ISBN: 978-981-11-2724-3; doi:10.3850/978-981-11-2724-3_0418-cd

Most of the differences relate to the level of detail of the models. For example, the application software is modelled in more detail in model checking than in PSA. From theoretical point of view, the main differences relate to modelling of time. Model checking enables much more flexible means for modelling time compared to the fault tree/event tree methodology (most commonly used approach for PSA).

The coupling between PSA and model checking can be loose or tight. In loose coupling, expert judgement is needed in the interface of the two methods. In tight coupling, no such judgement is required. Typically in tightly coupled approaches, an underlying algorithm of one methodology is utilized in the other methodology (e.g. to improve computational efficiency as in Bozzano et al. (2015a)).

In loosely coupled approaches, the methodologies are generally either used separately to analyze different aspects of the model, or the results of one methodology is used as input for the other. For example, in (Ortmeier et al. 2003), model checking is used to prove functional correctness with respect to a formal model, whereas fault tree analysis (FTA) is used to validate the model and to consider technical defects. The focus of our work is on loosely coupled approaches.

In Lahtinen and Björkman (2016a) several ways to utilize model checking and PSA together were identified. Most of them are loosely coupled approaches, and covered by the coupling approach presented in section 4.

3. Example System

Our example model is based on the DIGREL PSA model for a fictive boiling water reactor (BWR). The following brief description of the model is based on Authén et al. (2015).

The BWR has four redundant safety systems. The architecture of the safety I&C is presented in Figure 1. The protection system is divided into two subsystems, called RPS (reactor protection system) and DPS (diverse protection system). The I&C units included in the protection system are the acquisition and processing unit (APU) and the actuation logic unit (ALU). In addition, a manual control unit I&C unit (MU) is dedicated for operator actions.

The protection system is designed with fault tolerant features. The fail-safe actions are separately defined for each RPS/DPS safety function and for each actuation signal.

For model checking purposes, we constructed realistic but fictitious application software model using the Final Safety Analysis Report for the U.S. EPR nuclear power plant concept (Areva NP 2013) as a reference. The application logics (function block diagrams) for the actuation signals of the emergency core cooling (ECC) system and emergency feedwater (EFW) system were developed. Signal status processing was also included within elementary blocks. Signal status information refers to its assessed validity. The status information allows the application logic to, e.g., exclude invalid data from a majority vote. Status processing is used in different nuclear I&C system supplier's logics (Pakonen and Buzhinsky 2019).

The model consist of APU and ALU modules distributed between four redundant divisions and two subsystems. EFW is controlled by the DPS and the ECC is controlled by the RPS. The safety function processing logic for the RPS APU and ALU is shown in Figure 2. The safety function processing logic for the DPS APU and ALU is shown in Figure 3.

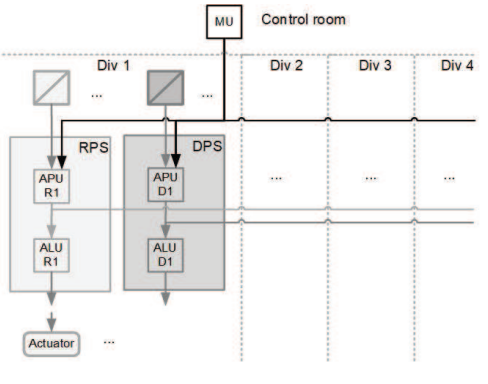

Fig. 1. Flow diagram of one train of the example NPP (modified from Authén et al. 2015). The architecture has been modified from (Authén et al. 2015). The main difference is that the measurements are not shared between divisions.

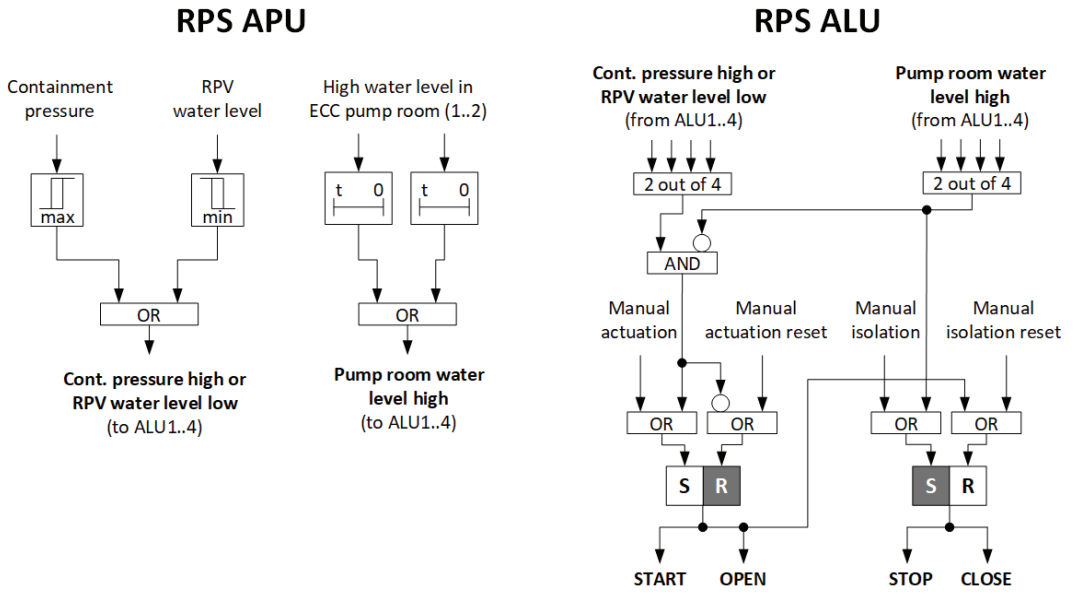

Fig. 2. The processing logic for each of the four redundant APUs and ALUs (RPS). The definitions for the function blocks can be found in Areva NP (2013).

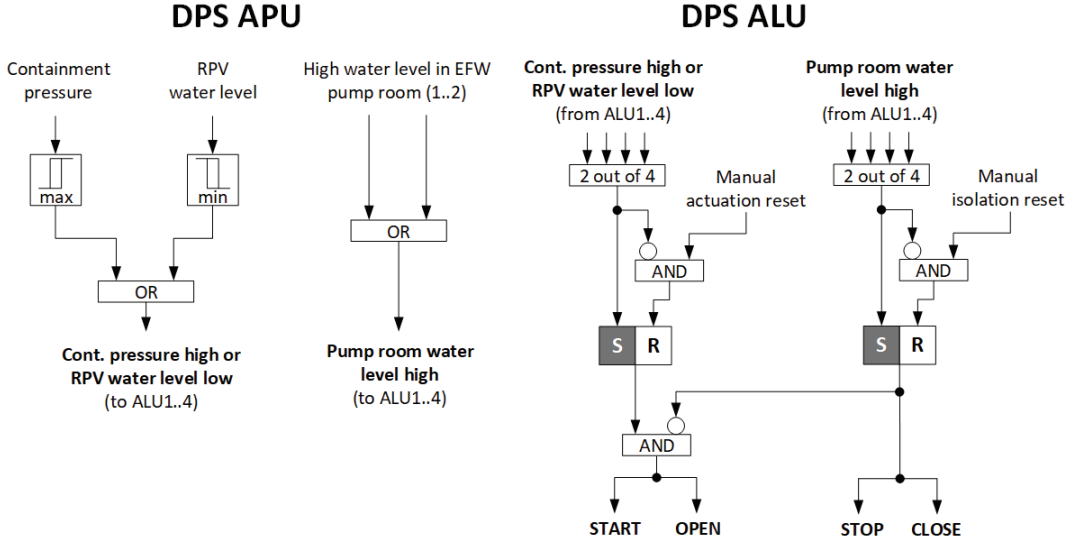

Fig. 3. The processing logic for each of the four redundant APUs and ALUs (DPS).

4. Coupling Approach

4.1 *Coupling model checking and PSA*

The general coupling approach is illustrated in Figure 4. The approach is based on the methodology presented in Lahtinen and Björkman (2016b). Mainly the approach for model checking has been updated and, thus, we focus in the approach description on that part of the methodology. For a more in-depth description of the other phases, we refer the reader to Lahtinen and Björkman (2016a, 2016b).

The main idea of the approach is that the model checking analysis is restricted to a smaller set of postulated hardware failures based on PSA results. Under these failures the system behavior is then analyzed more rigorously using model checking and the results are again compared against the PSA model.

First, we use PSA to compute the relevant minimal cut sets and related risk importance measures. The relevant set of minimal cut sets depends on the model checking scope. The cut sets can be computed, e.g., for a single fault tree representing a safety function, event tree accident sequence including several safety functions, or for a whole event tree. From model checking perspective, the basic events of a minimal cut set represent a combination of hardware (or software) failures events that are relevant for the analysis. Risk importance measures describes which basic events are most critical.

The model checking approach applied in Lahtinen and Björkman (2016b) was based on a methodology for modelling hardware failures described in Lahtinen (2014a, 2014b). In the development of the methodology, failure modes used in PSA where adopted. The applicability of the methodology for modelling hardware failures was studied using an older version of the DIGREL PSA model as a case study (Lahtinen 2014a). The case study demonstrated that the approach does not scale to large models consisting of several many redundant safety systems (failure behavior was not restricted).

Our early experiments with the current case study showed that the scalability did not improve considerably, even when the failure behavior of the model was restricted based on PSA results. Therefore, we adopted a new approach for the model checking phase.

4.2 *Model checking failure modelling approach*

Our alternative approach for model checking is based on work presented in Pakonen and Buzhinsky (2019). The approach uses a failure element, implemented as a code module for the NuSMV (Cimatti el al. 2002) model checker. Instances of the failure module are placed into locations in the model where failures are postulated to occur. At any time instant, the failure module can replace the output signal with a nondeterministic (Boolean or integer) value. The failure can be either self-announcing

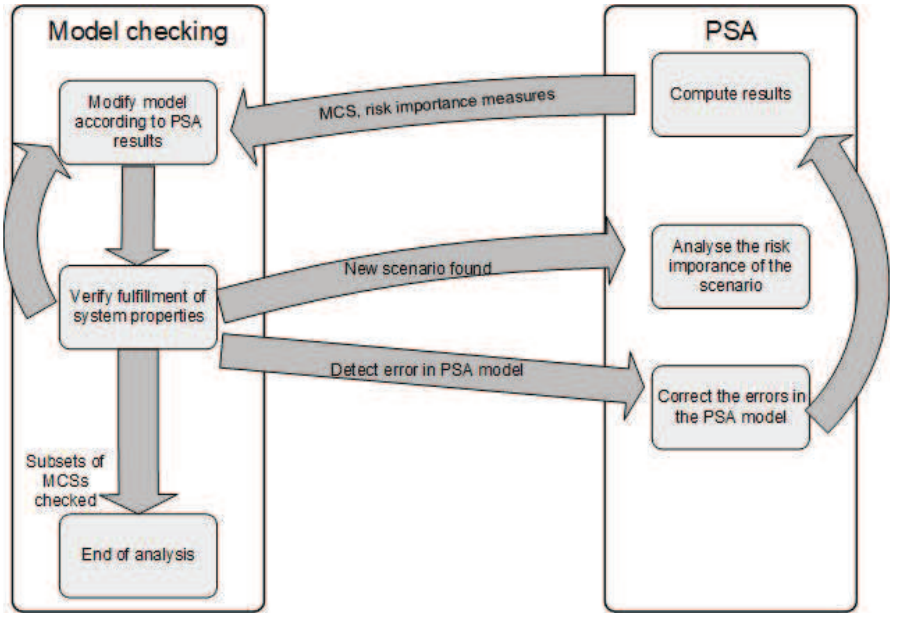

Fig. 4. Coupling approach (modified from Lahtinen and Björkman 2016b).

(apparent from the status of the signal) or non-self-announcing. In other words, the status of the signal is also non-deterministic upon failure. The true failure status is also given as an output, allowing the analyst to also observe failures that are non-self-announcing.

4.3 *Incorporating hardware failures*

To incorporate hardware failures, a failure module is simply placed into a "physical" location where a hardware failure is postulated to have an effect on a logic signal. We developed a set of rules to guide the placement of failure modules. The following hardware components are considered (as in Authén et al. (2015)):

- Processor module: A failure module is added after each output of the failed unit (APU/ALU).
- Digital/analog input module: A failure module is added for each input signal going to the digital/analog input module of the failed unit.
- Digital output module: A failure module is added after each relevant output of the failed unit.
- Communication module: A failure module is added at the communication link between the relevant units.
- Backplane: A failure module is added after each output of the failed unit.
- Power supply: A failure module is added after each output of the failed unit.
- Measurement: A failure module is added after the failed measurement.

In case of common cause failures (CCF), the failure modules are added for each redundancy present in the CCF. It is also possible to restrict the failure behavior (loss of function/spurious function and undetected/detected failure (Authén et al. 2015)) of a failure module. If we restrict the failure behavior scalability may improve, but the coverage of the analysis may decrease.

5. Case Study

We performed a case study using the graphical model checking tool MODCHK (Pakonen et al. 2017), which is based on NuSMV. For the case study, we included four model checking model configurations:

- C1: No failures are included in model.
- C2: The failure behavior is not restricted. Three redundancies of both systems can fail in an arbitrary manner.
- C3: The failures are limited to the output modules for ECC pump start and valve open signal, and EFW pump start and valve open signal of all ALUs. Failures are added for three redundancies.
- C4: The processor module of three redundant APUs can fail in both systems.

For each model configuration, we performed the same model checking analysis, i.e. identical properties were verified. We compared the analysis times (i.e. the time it for the model checker to verify the given property) and state spaces of the NuSMV model configurations.

For configuration C3 and C4, we used PSA results in the specification of failure locations. We selected for both systems two sets of CCFs that are among the most critical hardware failures and affect the systems in different ways.

According to Authén et al. (2015), the success criterion for EFW system and ECC system (when credited) is that one out of to four redundancies is able to perform its safety function. Thus, we added failures to three redundancies of both systems. Configuration C1 is included in the comparison to demonstrate the impact of the failure modelling to the analysis time and to the state space. For configuration C4, three different versions were modelled: (C4.1) the failure behavior of failure modules is not restricted, (C4.2) all failures are detected and (C4.3) all failures remain undetected. The purpose of the different versions is to study how detectability affects the analysis results. The following properties were analyzed:

- P1: Either ECC or EFW system shall be started (in at least one redundancy) when limit values of relevant measurements are reached.
- P2: There shall be no contradictory outputs.
- P3: Water leakage in a pump room shall stop the respective pump.
- P4: There shall not be any deadlocks.

For properties P2 and P3, the correct behavior is checked only for the redundancy that does not contain any failures. For property P3, we specified separate properties for ECC and EFW systems. The combined analysis time of both properties is measured. We split property P4 into eight sub-properties, four sub-properties for each system. We checked that it is always possible (under any initial condition) to reach a state in which the system is started, the system is not started, the system is stopped, or the system is not stopped. The combined analysis time of all sub-properties is measured. The results of the analyses are collected into Table 1.

Table 1. Summary of model checking results. The state space shows the number of reachable states. Computation times are given in seconds (under t(s) column). 'y' property was satisfied, 'n' property was not satisfied (under y/n column). '-' could not be computed in reasonable time.

Configuration	C1		C2		C3		C4.1		C4.2		C4.3	
State space	6.0E+47		2.5E+145		4.1E+58		9.0E+58		9.0E+58		4.8E+58	
time/satisfied	t(s)	y/n	t(s)	y/n	t(s)	y/n	t(s)	y/n	t(s)	y/n	t(s)	y/n
Property P1	2	y	-	-	3	y	3	n	1	y	3	n
Property P2	2	n	-	-	3	n	3	n	2	n	2	n
Property P3	3	n	-	-	4	n	4	n	4	n	5	n
Property P4	3	y	-	-	9	y	3	y	3	y	8	y

For configuration C4, the results differed a bit compared to the other configurations. Due to three undetected failures, either ECC or EFW system cannot be started. If the output of APUs is zero due to processor failure, only one of the inputs to the 2 out of 4 voting logic blocks can be set. Based on PSA results, two detected DPS APU processor failures should prevent EFW from starting and, thus, property P1 should not be satisfied for configuration C4.2 (ECC fails due to a design error in the RPS processing logics). The unexpected model checking results depend on how fault tolerant features are accounted for in the different models. The PSA model follows the principles described in Authén et al. (2015). Due to the default values specified for the safety function, two detected APU failures will cause a spurious stop signal. In the NuSMV model, the signal status processing will exclude the invalid data from the majority vote and, thus, two detected failures will not cause a spurious close signal.

The results clearly show that including all possible failures simultaneously in the model will complicate the model considerably. Already with a rather simple model, the state spaces grow almost by 100 orders of magnitude. None of the properties of the unrestricted failure model could be analyzed in reasonable time (under eight hours).

The number of states depends on the number and type of input signals and internal variables, but a state is only called reachable if there is at least one execution in which it appears (Bérard et al. 2001). The failure modules not only add variables, but also allow the model to reach states that it otherwise would not. Thus, the used modelling approach has an impact on the state space (e.g. the impact of time related function blocks is briefly discussed in Björkman et al. (2009)). We would like to note that the number or reachable states is not an unambiguous measure for system complexity. However, it can give an approximate indication.

By focusing the analysis on a certain set of hardware failures (based on PSA results), considerable improvement in computational times was achieved. For all of the restricted model configurations, the computation times were quite close to the model configuration with no failures.

The type of the hardware failures can have a large impact on the analysis results. For example, undetected failures and detected failures can cause a very different result. Whether the failure is loss of function or spurious function is of major importance, as well.

6. Discussion

The results of the case study demonstrate that the analysis time decreases considerably when the failure behavior is restricted. PSA results were used to specify critical failure points in the system architecture and the failure modules were added into those positions. The failure modules also enable restricting the type of a hardware failure. This could further improve the scalability of the approach.

In practice, it might be necessary to restrict the failure behavior of failure modules when model checking analysis goes beyond single failure tolerance analysis. Analyzing failure scenarios the system is not designed to handle is not practical or sensible as seen in the case study. For example, three undetected failures can prevent required actuation, whereas three detected failures cannot. For restricting the behavior, PSA results can be used. For the example system, the PSA results show that three undetected APU processor failures in one system will prevent the start of all four redundancies. Simplified failure modules could also be created to represent the restricted failures to facilitate the modelling and property specification.

Another benefit of the used model checking approach is that it does not specify what the failure modules represent. In the case study, we used the failure modules to represent certain types of hardware failures. The approach could also

enable to model, e.g., the software failure modes specified in Authén et al. (2015).

The limitations include the restricted analysis scope and the manual labor needed to build the models. The focus is to analyze a restricted set of hardware failure combinations selected based on PSA results, e.g. based on top 50 minimal cut sets. We note that it is not guaranteed that such an analysis scope is sufficient. It is possible that individual failures that do not occur in the most important minimal cut sets can also, together with a software design error, cause potentially unsafe scenarios. However, analyzing a larger set minimal cut sets may not be any longer practical. Only in case of very small systems, it may be feasible to analyze all combinations of failures exhaustively.

The current version of MODCHK does not support automatic generation of models based hardware failure combinations. Therefore, the models need to be built manually for each hardware failure combination. The process need to be automated, otherwise only a very limited set of hardware failure combinations can be considered. Our goal is to develop MODCHK to better support the analysis. Automating the process will make the coupling between PSA and model checking tighter.

7. Conclusions

There are many similarities between model checking and PSA. Coupling between PSA and model checking can be made on many levels. For example an underlying algorithm or data structured of one methodology can utilized in the other methodology (e.g. to improve computational efficiency), or the methodologies themselves are use together. The focus of our work has been in the co-use of the methodologies.

In this paper, we studied the applicability of a coupling approach by performing a case study. The approach has been developed for the verification of digital I&C systems including both hardware failures and detailed functionality of the I&C systems. The results of the case study showed that the scalability improves considerably, when the failure behavior is restricted. However, some limitations of the approach were also highlighted.

In the discussed approach, the coupling between model checking and PSA is loose, i.e. expert judgement is needed in the interface between PSA and model checking. When model checking and PSA are used together, the benefits of tight integration in practice are somewhat limited. This is partly due to the different levels of detail and how time can be considered in the models. Thus, loose coupling can be more applicable from a practical point of view, even though it may require quite a lot of manual labor. However, in many cases, at least part of the manual labor can be automated making the coupling more tight.

In our future work, we plan study the applicability of using model checking to support the definition of failure modes in the software failure modelling of PSA. Another loosely coupled approach we plan to study considers using PSA to support model checking in hardware failure tolerance analysis and I&C architecture assessment.

References

Areva NP (2013). U.S. EPR Final Safety Analysis Report. Tier 2, Chapter 7 Instrumentation and Controls, available at https://www.nrc.gov/reactors/new-reactors/design-cert/epr/reports.html referenced 17.10.2018. United States Nuclear Regulatory Commission, 2017.

Authén, S., J.-E. Holmberg, T. Tyrväinen, and L. Zamani (2015). Guidelines for Reliability Analysis of Digital Systems in PSA Context – Final Report, NKS-330, Nordic nuclear safety research (NKS), Roskilde.

Bérard, B., M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, and P. Schnoebelen (2001). Systems and Software Verification – Model-Checking Techniques and Tools, Springer-Verlag Berlin Heidelberg.

Björkman K, J. Frits, J. Valkonen, J. Lahtinen, K. Heljanko, I. Niemelä, and J.J. Hämäläinen (2009). Verification of safety logic designs by model checking. In 6th American Nuclear Society International Topical Meeting on Nuclear Plant Instrumentation, Control, and Human-Machine Interface Technologies, NPIC&HMIT 2009. Knoxville, Tennessee, 5 - 9 April 2009.

Björkman, K., J. Lahtinen, T. Tyrväinen, and J. -E. Holmberg (2015). Coupling Model Checking and PRA for Safety Analysis of Digital I&C Systems. in International Topical Meeting on Probabilistic Safety Assessment and Analysis, PSA 2015. American Nuclear Society, pp. 384-392., April 26 - 30, Sun Valley, ID, USA, 2015.

Bozzano M., A. Cimatti, A. Griggio, and C. Mattarei (2015a). Efficient Anytime Techniques for Model-Based Safety Analysis, In: Kroening D., Păsăreanu C. (eds) Computer Aided Verification. CAV 2015. *Lecture Notes in Computer Science*, vol 9206. Springer, Cham.

Bozzano, M., A. Cimatti, O. Lisagor, C. Mattarei, S. Mover, M. Roveri, and S. Tonetta (2015b). Safety Assessment of AltaRica Models via Symbolic Model Checking, *Sci. Comput. Program.* 98(4), pp. 464–48.

Chen, C., F. Zeng, and M. Lu (2015). A Verification Method for Software Safety Requirement by Combining Model Checking and FTA., in 2015

International Industrial Informatics and Computer Engineering Conference (IIICEC 2015) Xi'an, Shaanxi, China, January 10-11, 2015. 10.2991/iiicec-15.2015.301.

Cimatti, A., E. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R Sebastiani, R and A. Tacchella (2002). NuSMV Version 2: An Opensource Tool for Symbolic Model Checking, in International Conference on Computer-Aided Verification (CAV 2002), ser. LNCS, vol. 2404. Springer.

Clarke, E. M., O. Grumberg, and D.A. Peled (1999). *Model Checking*, The MIT Press 1999.

Lahtinen, J. (2014a). Verification of Fault-Tolerant System Architectures using Model Checking, in 1st International Workshop on DEvelopment, Verification and VAlidation of cRiTical Systems (DEVVARTS), Florence, September 8th, 2014, Volume 8696 of Lecture Notes in Computer Science, pp. 195-206, Springer.

Lahtinen, J. (2014b). Hardware Failure Modelling Methodology for Model Checking, Research report: VTT-R-00213-14, VTT Technical Research Centre of Finland. Available online: http://www.vtt.fi/inf/julkaisut/muut/2014/VTT-R-00213-14.pdf

Lahtinen, J. and K. Björkman (2016a). Feasibility study on the integration of PRA methods and model checking. Research report (VTT-R-04924-15), VTT Technical Research Centre of Finland Ltd., Espoo, Finland.

Lahtinen, J. and K. Björkman (2016b). Integrating model checking and PRA: A novel safety assessment approach for digital I&C systems. In L. Walls, M. Revie, & T. Bedford (Eds.), Risk, Reliability and Safety: Innovating Theory and Practice [383] CRC Press.

Ortmeier, F., G. Schellhorn, A. Thums, W. Reif, B. Hering, and H. Trappschuh (2003). Safety Analysis of the Height Control System for the Elbtunnel. *Reliability Engineering & System Safety* 81 (3): pp. 259 – 268.

Pakonen, A. and I. Buzhinsky (2019). Verification of fault tolerant safety I&C systems using model checking, in 20th International Conference on Industrial Technology (ICIT2019), 13-15 Feb 2019, Melbourne, Australia, pp. 969–974.

Pakonen, A., T. Tahvonen, M. Hartikainen, and M. Pihlanko (2017). Practical Applications of Model Checking in the Finnish Nuclear Industry, in 10th International Topical Meeting on Nuclear Plant Instrumentation, Control and Human Machine Interface Technologies (NPIC & HMIT 2017), 11 - 15 June 2017, San Francisco, CA, USA, pp. 1342-1352.

Volk, M., S. Junges, and J. Katoen (2018). Fast Dynamic Fault Tree Analysis by Model Checking Techniques. *IEEE Transactions on Industrial Informatics*, vol. 14, no. 1, pp. 370-379, Jan.