

VTT Technical Research Centre of Finland

Demonstration of a Model-based Approach for Formal Verification of I&C Logics

Linnosmaa, Joonas; Pakonen, Antti; Alanen, Jarmo

Published in:

13th Nuclear Plant Instrumentation, Control and Human-Machine Interface Technologies (NPIC&HMIT 2023)

DOI:

doi.org/10.13182/NPICHMIT23-41121

Published: 01/07/2023

Document Version

Publisher's final version

[Link to publication](#)

Please cite the original version:

Linnosmaa, J., Pakonen, A., & Alanen, J. (2023). Demonstration of a Model-based Approach for Formal Verification of I&C Logics. In *13th Nuclear Plant Instrumentation, Control and Human-Machine Interface Technologies (NPIC&HMIT 2023)* (pp. 1441-1450). American Nuclear Society (ANS).
<https://doi.org/doi.org/10.13182/NPICHMIT23-41121>



VTT
<http://www.vtt.fi>
P.O. box 1000FI-02044 VTT
Finland

By using VTT's Research Information Portal you are bound by the following Terms & Conditions.

I have read and I understand the following statement:

This document is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of this document is not permitted, except duplication for research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered for sale.

Demonstration of a Model-based Approach for Formal Verification of I&C Logics

Joonas Linnosmaa^{1*}, Antti Pakonen², Jarmo Alanen¹

¹VTT Technical Research Centre of Finland Ltd, Tampere, Finland

²VTT Technical Research Centre of Finland Ltd, Espoo, Finland

doi.org/10.13182/NPICHMIT23-41121

ABSTRACT

This paper introduces a model-based methodology for conformity assessment of I&C logics using model checking analysis. The presented method extends our previous work of model-based, artefact-driven support for engineering of mission-critical systems. The approach includes an ontology and a data model for constructing a tool-supported data repository for the model checking artefacts. The repository brings to the assessment, among others, the benefits of traceability between requirements, claims and analysis results, and it acts as a sole source for information, avoiding distortion and fragmentation of data. For demonstrating and testing the capabilities of our approach, we performed an exemplary model checking task on an I&C related case study, storing all the created work items to the data repository created with Siemens' Polarion™ tool. Finally, we present a methodology for using the relations between the artefacts to automatically generate assessment reports and explore the capabilities of the selected tool for this task.

Keywords: MBSE, Model checking, I&C assessment

1. INTRODUCTION

The nuclear industry has traditionally relied on a document-centric engineering approach. With non-structured documents, manual copying and manipulation is needed in passing information between engineering and assessment processes and tools. Model-Based Systems Engineering (MBSE) [1], on the other hand, is about formalized application of modelling principles, methods, languages, and tools for the entire life cycle of complex systems. The shift from document-centric to domain model-based information exchange paves the way towards an integrated engineering framework, promotes clarity, and facilitates formal verification of design solutions. Our goal is to model an engineering information repository.

Model checking [2] is a formal verification method that can be used to mathematically prove that a model of a (hardware or software) system fulfils given (formal) properties. In the Finnish nuclear industry, the method has been successfully used for over a decade to analyze instrumentation and control (I&C) logics. One practical challenge is the lack of tool support for linking (1) the functional requirements with the formal properties, and (2) the verification results with safety claims.

The approach presented here is based on authors' previous systems engineering ontologies (called Systems Engineering Artefacts Model (SEAModel) [3]), which are meant to support the MBSE of critical systems by providing a model for the repository of system engineering artefacts. In our earlier work, we have developed a hybrid (supporting safety, security and dependability) system engineering ontologies [4] and data models

*joonas.linnosmaa@vtt.fi

[5] [6] for supporting the MBSE of critical systems. An ontology models the semantics of domain concepts, allowing computers to automatically search, refine and combine pieces of distributed information. Previously, in [4], the SEAModel was enhanced to cover not only safety, but security and dependability engineering, and a specific security risk analysis method (STA). As in [6], the goal of the model-based approach to provide a gapless traceability loop from the conformity assessment claims to the requirements that are claimed to be satisfied, and to the determination reports, which are used as the evidence for the claims. The traceability loop continues from the determination reports to the design or implementation artefacts under assessment, and from the design and implementation artefacts that are claimed to satisfy the particular requirements to those requirements. In this paper, we further test the generalization and the extendability of the SEAModel conformity assessment to cover another analysis, the model checking.

2. MODEL-BASED METHODOLOGY

2.1. Approach

In this work, we model a repository of a systems engineering artefacts, i.e., the work products that are relevant to the conformity assessment process. To set up a data repository-supported demonstration of model-based approach for formal verification of I&C logics using model checking analysis the following was done:

- We extended the existing semi-formal SEAModel conformity assessment ontology to include new artefacts needed for model checking analysis. This is further explained in Subsection 2.2, and the resulting ontology is shown in Fig. 1.
- We created a data model for the model checking artefacts from the extended conformity assessment ontology. The data model is shown in Fig. 2 and is further explained in Subsection 2.3.
- We created a structured data repository using Polarion REQUIREMENTS software tool, including ready made templates for all the work items. This step is further explained in Subsection 2.4.
- We specified a semi-fictitious industrial case of a reactor protection system, mimicking roughly the PS of the proposed U.S. EPR, and performed an exemplary model checking analysis. The created work items were stored to the data repository with all the attributes and traceability links. This step is also further explained in Section 3.
- We studied and demonstrated automatic report generation with the chosen tool. This step is further explained in Section 4.

These steps aim to support the conformity assessment with model checking analysis by linking the functional requirements with the formal properties and the analysis results with safety claims. The selected demonstration tool was only used to demonstrate and evaluate the SEAModel; the purpose of this work was not to evaluate the tool itself. Our models are meant to be tool-agnostic, meaning that they do not purposefully contain any tool-specific modelling decisions, and it is possible to configure them to work in any suitable data repository or life cycle management tool.

2.2. Semi-formal Ontology for Model Checking

Model checking [2] is a formal verification method, where a software tool called a model checker is used to determine if a model (of, e.g., a software element or a hardware element) satisfies stated formal properties. In case the model checker finds a model execution that is contrary to a property, the execution is returned to the user as a counterexample scenario. The counterexample can then reveal a design issue, or an error in the way the analyst specified either the model or the property. In Finland, model checking has been used to verify I&C application logic design in the nuclear and railway industries. In the nuclear industry alone, dozens of design issues have been identified, often in designs that had already been subjected to other verification methods [8]. One practical challenge in the application of model checking is the limited tool support. For example, specifying the formal properties based on the (usually textual, nature language)

functional requirements is a manual task. The utilisation of the results (including the counterexamples), in either refining the design or producing a verification report, is not systematic. Overall, poor traceability is an issue.

To adopt model checking analysis as a part of the SEAModel approach, the conformity assessment ontology from [3] was extended to include the needed model checking analysis artefacts. More accurate description of the SEAModel ontology can be found from the reference, it won't be explained in detailed in this paper. One important conformity assessment core concept is the traceability loop, the combination of requirement, engineering artefact, determination result, evidence and claim. Structured linking between claims, reasoned arguments and with evidence that justifies the claim, supplies the assurance cases with many desirable properties such as traceability, assessability and manageability. As explained in [6] we optimized the assurance case structure for easy implementation in a database-oriented tool. Optimization is accomplished by including the Argument element into the Claim element as one of its attributes; this is justified by the fact that typically the relationship between Claim and Argument is a one-to-one relationship and they have the same lifecycle.

The extended ontology is presented in Fig. 1 as a semi-formal UML class diagram. It can be seen that all the model checking artefacts are specialized conformity assessment artefacts, and thus no new core artefacts were needed. The colours of the figure are meant to help the reader: blue objects represent generic system engineering artefacts, red objects represent individual conformity assessment-related artefacts, dark green represent conformity assessment-related artefacts and the light green objects are the new model checking-related artefacts.

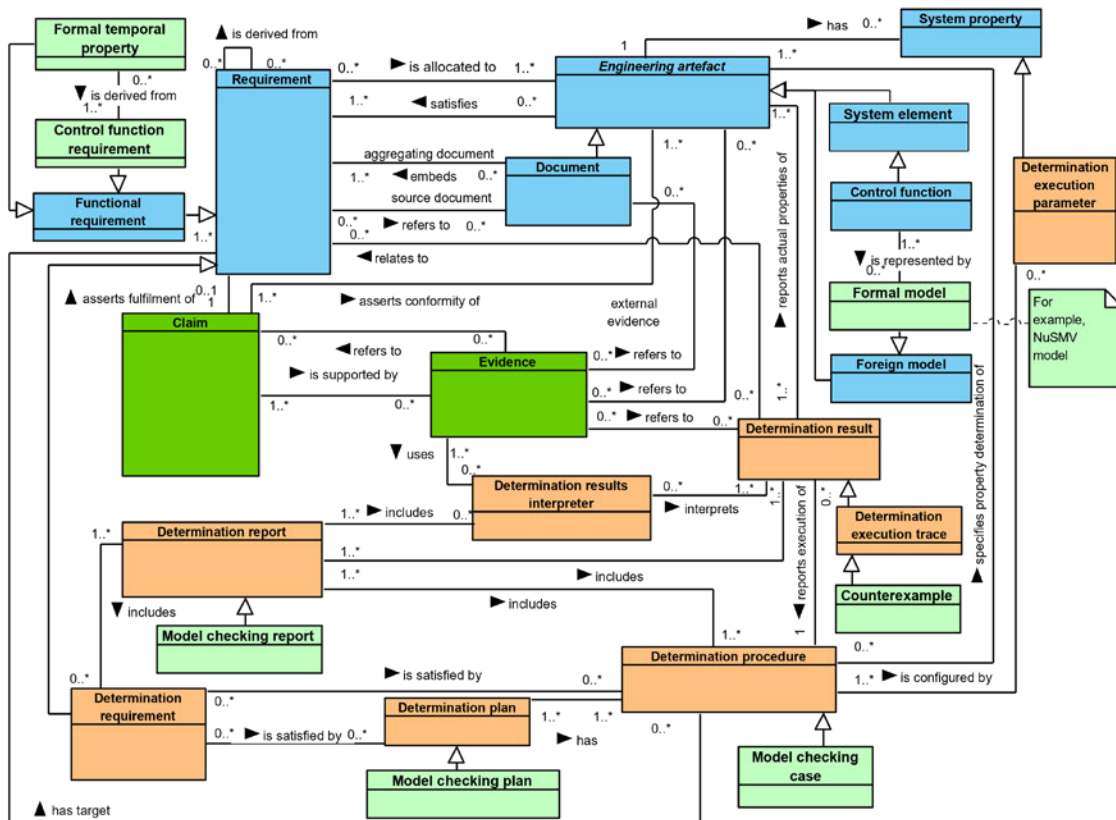


Figure 1: Mapping of the model checking ontology to the conformity assessment ontology

It should be noted, that in this paper we do not define a model checking analysis procedure, but only an ontology for the input and output artefacts. This ontology (and the derived data model) is meant to be used

in the context of the analysis procedures. The objective of this semi-formal ontology is to provide a model for a structured systems engineering repository. However, not all the model elements (classes) are meant to be strictly implemented as individual entities, e.g. data tables, but the ontology helps to derive the schema and the traceability links for the repository.

2.3. Datamodel for Repository Supported Model Checking Conformity Assessment

From the general conformity assessment ontology (extended with model checking) presented in Fig. 1 to a data repository oriented implementation, we derived a data model for the model checking analysis. The data model describes the needed data object types, their attributes and interrelations for the tool supported data repository. Data model is helpful, but not necessarily a required, step for creating the tool-supported data repository. It defines all the needed data object types and their relations to each other. Fig. 2 shows the created work items for the case study. It is similar to the general ontology, only omitting some general artefacts and using the needed specializations instead. The colours again follow the same schema as earlier: blue objects represent generic system engineering artefacts, red objects represent individual conformity assessment-related artefacts, dark green represent conformity assessment-related artefacts and the light green objects are the new model checking-related artefacts.

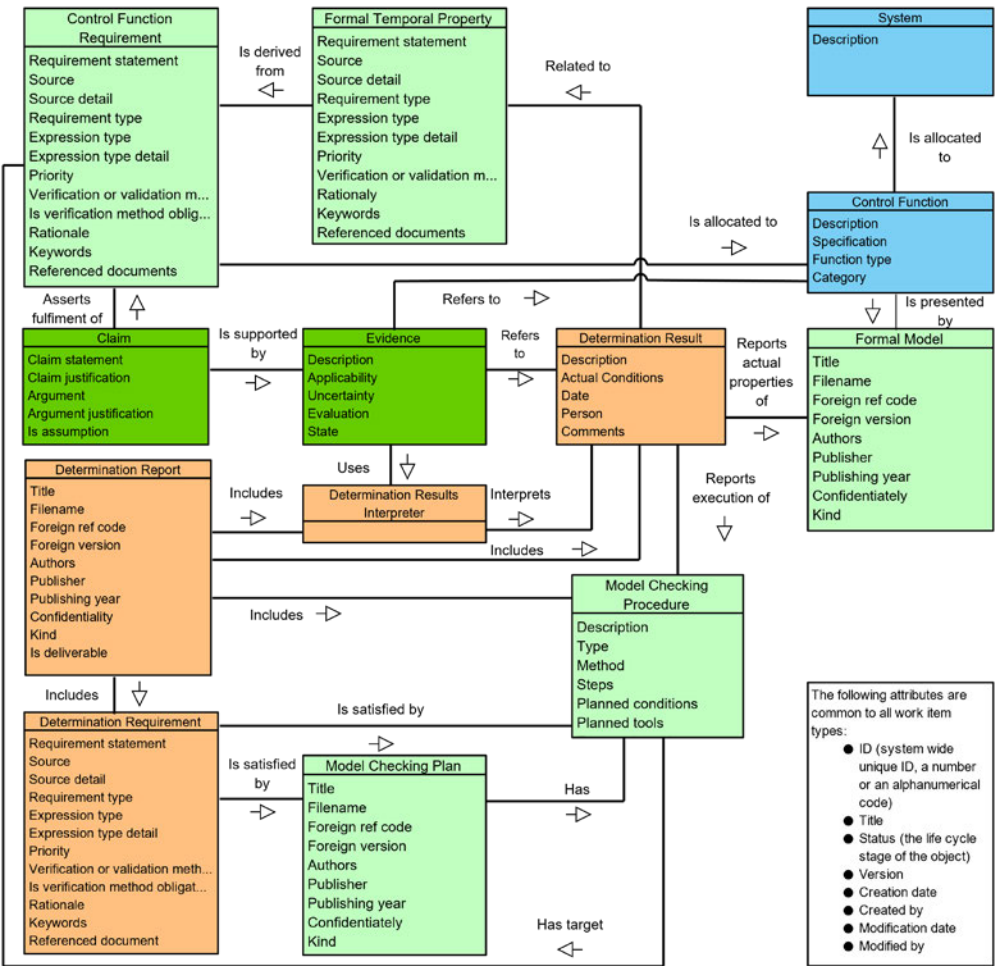


Figure 2: Model checking data model.

Each of the data model objects also contain a set of attributes (some of them are also visible in Fig 2). White box list attributes common for all the elements. For each artefact and attribute of the model, an

explanation of it's meaning is provided, and the data types of the attributes have been specified (such as textual, numerical or list etc.). Unlike in the ontology in Fig 1, the number of artefact instances is not specified in the data model as commercial off-the-shelf tools do not usually support implementation of relationship cardinalities.

2.4. Model Checking Data Repository Implementation

To further support the model-based approach for model checking, we created a structured data repository supported by a life cycle management tool, that will host the work items created during the analysis work. As explained above, the basis for this repository was the model checking data model from Fig 2. The data model can be implemented using many data repository oriented tools, but it is preferred to use a tool that can implement relations between data objects as traceable links as that can support impact analysis. For the demonstration in this paper, we used the Polarion® REQUIREMENTS™ requirements management tool, which is a well-established industrial tool.

What we consider as a data object template is an empty object with specific preconfigured data fields available for inserting the data created during the analysis work. The same template can be used to create as many independent object as are needed. We have worked with Polarion doing similar SEAModel repositories before, for example in [6,4]. Thus, we followed a similar workflow to create the repository, it comprised of following (Polarion specific) steps: 1) Created the needed work item types (representing artefacts/classes from the data model. 2) Created custom fields for these Work Items (representing attributes of the artefacts/classes). 3) Created enumerations and dependent enumerations for custom fields (these are predefined selection lists). 4) Created link roles and rules for work items (named traceability links and their controls). The only artefact from Fig. 8 which we did not implement during this work as a template, was the 'Document' artefact type. Polarion supports semi-automatic document generation (using Polarion LiveDoc), which we studied to be used to implement the semi-automatically 'Document' artefacts.

Doing this, we created the elements from the model checking data model as ready-to-use work item templates in Polarion, with most of the attributes available as fields, and link roles and rules according to the data model available. The tool will act as a repository for the work items created during the assessment.

3. CASE STUDY: I&C MODEL CHECKING PROJECT

3.1. Motivation

The tools and the work process used in practical model checking projects in Finland is described in [8]. Activities that could still benefit from further tools support include:

1. Requirements elicitation, where the analyst collects functional requirements from different sources, that can also include manuals or other system descriptions in addition to proper requirement specification documents.
2. Formal property specification, where the analyst translates the requirements into temporal logic formulae, using languages like Linear Temporal Logic (LTL) [2] and Computation Tree Logic (CTL) [2].
3. Documentation, where the reports need to describe, among other things, (1) the systems, functions, and requirements that were analyzed, (2) the verification results for each analyzed requirement (true/false), (3) any design issues that violate a requirement. (The presentation of the design issues is derived from the counterexamples produced by the model checker).

To gain confidence on the feasibility of our model-based approach to model checking, we tested it with an nuclear instrumentation and control (I&C) related example case.

3.2. Case example

For our case study, we envisioned a project where an analyst uses model checking to verify the logic of the Protection System (PS) of the proposed U.S.EPR [9]. While some elements shown in Fig. 4 (e.g. the name

of the I&C system, and the references to source documents) are based on the U.S.EPR Final Safety Analysis Report [9], other elements (the requirements, formal properties, and the logic diagram for the fictitious PS function “SF-01”) are mostly based on invention (see [10]).

In the envisioned project, the analyst models the SF-01 logic, formalises properties based on the functional requirements for SF-01, uses the NuSMV [11] model checker to verify the model against the properties, and then writes a report. The activities follow a plan. A counterexample produced by NuSMV describes a violation of a certain requirement about the manual reset of a trip signal.

3.3. Polarion Work Items

During the study, based on the envisioned case and the model-based approach, we did a model checking analysis as a part of conformity assessment and stored the created artefacts in the created Polarion data repository as work items. Work items include the needed attributes and traceability links to other work items according to the specified data model (from Fig. 2). Fig. 3 shows the linked work items for the evidence work item ‘S004’. As shown in the figure and defined in the ontology and the data model, evidence artefacts can be linked to four other artefacts; claims, determination results, control functions and to determination result interpreters. These links are used to perform impact assessment and automatically flag work items in case there is a change in a linked work item. For example, a claim might be suspected when a evidence that claim is supported by is changed and thus the argument might not be valid anymore.

|  Linked Work Items | | |
|---|-----------|--|
| Suspect | Role | Title |
| | refers to |  S010 - Determination Result - Violations of requirement S001 |
| | refers to |  S012 - Engineering Artefact/System Function/Control Function - Manual Reset |
| | uses |  S006 - Determination Result Interpreter - Model checker interpreter |
| | supports |  S002 - Claim - Req S001 |

Figure 3: Polarion linked work items

Just to show easier what was deposited in the repository, we present Fig. 4, which shows the created case study work items, their relations, and some selected attributes relevant to them (again according to the data model from Fig. 2).

4. AUTOMATIC REPORT GENERATION

4.1. Fetching the Traceability Loop from the Repository

As explained in the previous sections, in the heart of our model-based conformity assessment is the traceability loop between the requirement, engineering artefact (in this case, the control function), claim, evidence and the determination result. The following traceability example builds upon the fact that a data repository exists where the artefacts have been created during a model checking analysis and conformity assessment and explains how an assessor might use it to fetch the needed parts of the loop from the repository to be used in reporting or otherwise reviewing the assessment results. Fig. 5 explains the same idea with the help of the relations from the ontology.

First, the target system of the verification, is selected. Among the control functions allocated to that system, the logics which were verified are selected. Based on the selected control functions, a list of requirements allocated to that system should be generated. A formal temporal property should have been created for those requirements which were assessed using model checking. If a formal temporal property has been verified through a model checking procedure, a determination result should exist in the repository linked to the

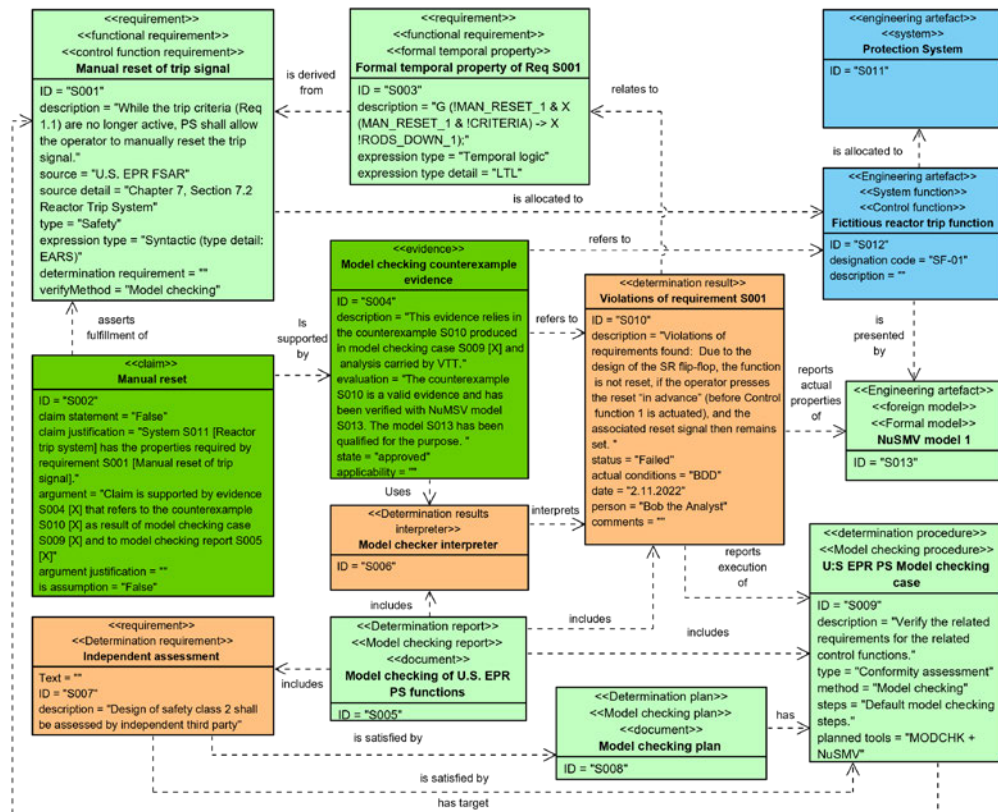


Figure 4: Model Checking example work items

property. The determination result (the counterexample) should be referred by an existing evidence artefact. Finally, a claim should be found supported by this evidence, which asserts the fulfilment of the original requirement. This is only one way to traverse the repository through the traceability links for accessing the conformity assessment artefact, there are multiple ways to explore the linked artefacts, depending on the wanted start point and the needed artefacts.

To document the results of a conformity assessment using model checking, for example to a customer, usually a model checking report is written. The report should show the fulfilment of the assessed requirement by claims and the show credible evidence. Often, following a document-based approach, the assessor starts writing a word document referencing and copying their assessment result and analysis findings, usually stored in some other document or output files generated by their analysis models or tools. Before finalizing the report, the assessor must ensure that all the collected information is up-to-date with respect to the final work done. On the other hand, the repository supported model-based approach, presented in this paper, is designed to support semi- or fully automatic document generation. How automatic the final generation is, depends on the tool used for building the document and the possibilities offered by it. The generation is based on the traceability loop and fetching the artefact from the repository using the links build within the database, for example using the workflow presented above.

4.2. Creating a Document Template

Based on the model based approach, generating new (and new versions of) assessment reports should be simple, and up-to-date artefacts should be possible to be fetched from sole source of information, a structured repository. Which parts of the traceability loop are needed to be documented or reported, depends on the needs of the assessor and required report. For easy generation of documents, for example, a template for a wanted document format should exist, where the needed information can be fetched from the repository.

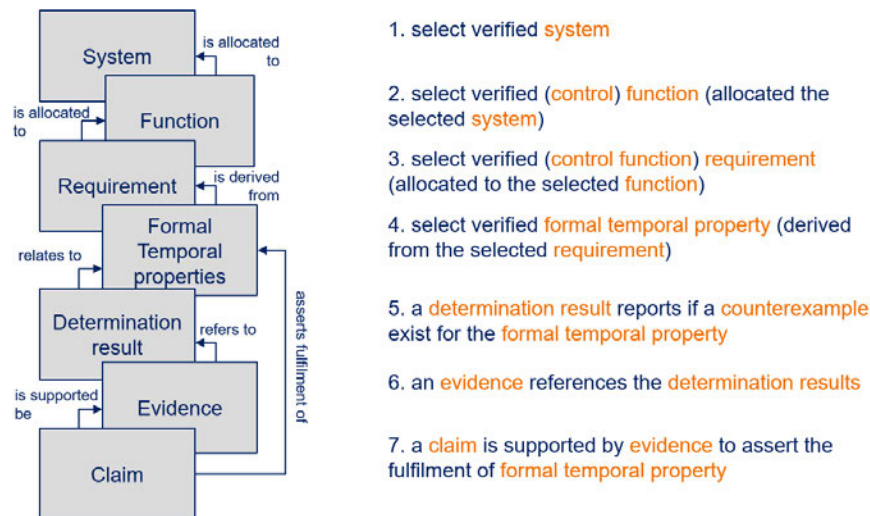


Figure 5: Workflow for fetching the traceability loop

For generating a generic (semi)automatic report for model checking, the following tool-agnostic workflow could be used.

A repository has been prepared with templates available for the model checking artefacts (for example, as explained in Section 2). A document template is created with all the static parts premade (parts that do not change between projects, documents or assessments), and placeholders are specified for the dynamic parts (project specific artefacts created during the assessment, which will change between documents/assessments), which can be automatically updated when the assessment is done and the document is needed to be generated. The model checking analysis has been performed (the validity of certain claims is verified) and the created artefacts are implemented as a data repository work items with the needed attributes and relation links to other work items. The needed parts of the traceability loop are fetched to dynamic parts of the document (for example, as explained above). The report document is generated, using the specified document template, using the static parts and populating the dynamic parts of the document with the created work items automatically.

4.3. Document Generation with Polarion

To test and demonstrate the model-based approach, we used the Polarion LiveDocs for the semi-automatic generation of the assessment report. The way was selected as the data repository was already built using Polarion Requirement, and the LiveDocs integrate to the database, allowing fetching of work items to the documents. LiveDocs are special document-like containers for both free-form text and granular work items. For example, in a requirements specification document there is a possibility to mark content elements as Requirement-type work items that can be linked for traceability, workflow controlled, assigned, planned, scheduled, reported, and managed throughout the artifact lifecycle [7].

For the demonstration, we tested the ability of LiveDoc to fetch the needed case study information from the model checking work items in the Polarion Repository to an exemplary assessment report. As explained under the case study, the report need to describe, among other things, (1) the systems, functions, and requirements that were analyzed, (2) the verification results for each analyzed requirement (true/false), (3) any design issues that violate a requirement. (The presentation of the design issues is derived from the counterexamples produced by the model checker). Creating a document template for the model checking report using Polarion was quite straightforward, and this paper won't focus on creating the static parts of the document, but on fetching the work items from the repository. Basically this meant bringing the needed information from the work items showed in Fig. 4 to the document and creating the needed traceability links

between them.

Bringing the information was also straightforward, and Fig. 6 shows the document version of the Claim S002 work item fetched to the document from the repository. With the work item, we have fetched the relevant attributes related to it: claim statement, claim justification, argument and the linked work items. This way, the information is not needed to be copied from anywhere, but it is imported from the repository. What attributes are actually needed in the final report is of course depending on the assessor and the customer, the tool supports the selection of needed parameters. Similar approach was followed for the other work items; requirement, evidence and the determination result. Other desired aspect of our approach is the linking of the document and the work items presented in it. This would extend the automatic suspecting and impact analysis of changes in the work items to also cover the documents (between work-items this already happens in Polarion). Based on the demonstration, the work items imported to the document are connected to the document, and can be traced to it by checking the status of the work item through Polarion user interface.

4.2 Verification results

We found violations of requirements in SF-01.

S002 - Claim - Requirement S001 is fulfilled through model checking analysis



| | |
|----------------------------|---|
| Claim statement | False |
| Claim justification | System S011 [Reactor trip system] has the properties required by requirement S001 [Manual reset of trip signal]. |
| Argument | Claim is supported by evidence S004 [Model checking counterexample] that refers to the counterexample S010 [Violation of requirement S001] as result of model checking case S009 [U.S EPR PS Model checking case]. |
| Linked Work Items | is supported by:  S004 - Evidence - Model checking counterexample , asserts fulfillment of:  S001 - Requirement/Functional Requirement/Control Function Requirement - Manual reset of trip signal |

Figure 6: Screenshot of automatically populated claim work item in the generated Polarion LiveDoc

Thirdly, our approach pursues the automatic generation of the document, which would be based on the relations between the artefacts from the ontology. In the ideal case, it would be enough to select, for example, the claim, and the other related artefacts traceability loop (such as evidence, determination result, requirement) could be fetched to the document automatically (following some convenient database query). However, as it was tested during the demonstration it was not supported by our selected tool.

5. CONCLUSIONS

In this work, we presented a model checking analysis conformity assessment ontology, a related data model and tested their capabilities by building a tool-supported data repository. In addition, we performed a nuclear related I&C model checking analysis as an exemplary case study and stored the model-checking analysis artefacts to a repository created with Siemens' Polarion™ Requirements with all the attributes and traceability links as specified in the data model. We then successfully generated the traceability loop of a conformity assessment report semi-automatically using the data repository tool by importing the information from the created work items to a document template, which could be then used by the assessor to report the results of the assessment.

Through the demonstration we gained confidence that the data-model and the structured artefact repository can bring many wanted properties to the assessment process, which are often prevalent with non-structured

document-based approaches. The benefits include providing a sole source of information for reports and avoiding the need to copy and paste the same information to several documents (which often leads to fragmentation and distortion of information following from different revisions of data). Other big benefit is the traceability of the model checking artefacts, meaning the determination results can be traced back to procedures and models used for the analysis to the requirements checked. This also supports the impact analysis of changes, automatically flagging the need of re-verification in the case of changing design or requirements.

In the future, we intend to extend the methodology to further cover impact tracking and the automatic flagging of the artefacts and documents as new revisions or updates are generated for the existing artefacts. We also aim to add automation steps for storing the assessment results to the repository automatically as they are created through analysis models and tools.

ACKNOWLEDGEMENTS

The Finnish Research Programme on Nuclear Power Plant Safety 2019–2022 (SAFIR2022) funded this research. Any opinions or findings of this work are the responsibility of the authors, and do not necessarily reflect the views of the sponsors or collaborators.

REFERENCES

- [1] A. L. Ramos, J. V Ferreira, and J. Barceló, “Model-Based Systems Engineering: An Emerging Approach for Modern Systems“, *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, **42**(1), pp. 101–111 (2012).
- [2] E.M. Clarke, O. Grumberg and D. Peled, *Model checking*, MIT press, Cambridge, MA (1999).
- [3] T. Tommila and J. Alanen, “Conceptual model for safety requirements specification and management in nuclear power plants“, *VTT Technology* 238 (2015).
- [4] J. Alanen, J. Linnosmaa, T. Malm, N. Papakonstantinou, T. Ahonen and E. Heikkilä, “Hybrid ontology for safety, security, and dependability risk assessments and Security Threat Analysis (STA) method for industrial control systems“, *Reliability Engineering & System Safety*, **220** (2022).
- [5] J. Alanen, J. Linnosmaa and T. Tommila, “Conformity assessment data model“, *VTT Research Report*, VTT-R-06743-17 (2017).
- [6] J. Linnosmaa and J. Alanen, “Demonstration of a conformity assessment data model“, *Proceedings of 17th International Conference on Industrial Informatics, INDIN 2019*, Helsinki, Finland, 22-25 July 2019, pp. 369-373 (2019).
- [7] Siemens Industry Software, Inc., “Polarion: Administration’s Guide“, https://docs.plm.automation.siemens.com/content/polarion/19.3/help/en_US/user_and_administration_help/administrators_guide.html (2023).
- [8] A. Pakonen, I. Buzhinsky and K. Björkman, “Model checking reveals design issues leading to spurious actuation of nuclear instrumentation and control systems“, *Reliability Engineering & System Safety*, **205**, 107237 (2021).
- [9] Areva NP, U.S. EPR Final Safety Analysis Report, <https://www.nrc.gov/reactors/new-reactors/design-cert/epr/reports.html> (2013).
- [10] I. Buzhinsky and A. Pakonen, “Model-Checking Detailed Fault-Tolerant Nuclear Power Plant Safety Functions“, *IEEE Access*, **7**, pp. 162139-162156 (2019).
- [11] A. Cimatti, E. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani and A. Tacchella, “NuSMV 2: An Open Source Tool for Symbolic Model Checking“ *2002 International Conference on Computer Aided Verification (CAV)*, LNCS 2404, pp. 359-364 (2002).