Case Study on Automated and Continuous Reliability Assessment of Software-Defined Manufacturing Based on Digital Twins

Philipp Grimmeisen
philipp.grimmeisen@ias.uni-stuttgart.de
University of Stuttgart
Stuttgart, Germany

Andreas Wortmann
wortmann@isw.uni-stuttgart.de
University of Stuttgart
Stuttgart, Germany

Andrey Morozov
andrey.morozov@ias.uni-stuttgart.de
University of Stuttgart
Stuttgart, Germany

ABSTRACT

Traditional production systems are characterized by rare software updates and fixed production lines. Each production unit is designed and programmed for a specific task. Therefore, the reliability assessment is conducted once before the operation, mostly manually, and is based on traditional reliability models, such as event trees, fault trees, or reliability block diagrams. In comparison to traditional production systems, the focus of modern, complex production systems is shifted towards the software part. This is emphasized by the concepts of digital twins and Software-Defined Manufacturing (SDM). These software-intensive and safety-critical systems have more frequent software updates to address higher system flexibility and adjustable production processes. Therefore, SDM systems require a new approach to reliability assessment. Each software update can change the system behavior significantly. This leads to the necessity to reconduct the reliability assessment automatically before each software update. Advanced and hybrid reliability models are the key enabling technology. These models must be automatically generated and synchronized with the available system models and digital twins. Model-to-Model (M2M) transformation methods are another enabling technology.

In this paper, we present a case study on automated and continuous reliability assessment of SDM. It shows, that our new method is a suitable candidate to enable the reliability assessment of SDM based on digital twins. The method includes (i) the extension of SysML v2 for reliability assessment, (ii) the automatic generation of hybrid reliability models from the digital twin, and (iii) their reliability assessment with new solvers developed for our OpenPRA framework.

KEYWORDS

Case Study, Reliability Assessment, Software-Defined Manufacturing, Digital Twin, SysML v2, Model-to-Model (M2M) transformation

MODELS '22 Companion, October 23–28, 2022, Montreal, QC, Canada
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9467-3/22/10…$15.00
https://doi.org/10.1145/3550356.3561601

ACM Reference Format:
Philipp Grimmeisen, Andreas Wortmann, and Andrey Morozov. 2022. Case Study on Automated and Continuous Reliability Assessment of Software-Defined Manufacturing Based on Digital Twins. In *ACM/IEEE 25th International Conference on Model Driven Engineering Languages and Systems (MODELS '22 Companion), October 23–28, 2022, Montreal, QC, Canada.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3550356.3561601

1 INTRODUCTION

Software-Defined Manufacturing (SDM) [23] is a concept derived from information and communication technology Software-Defined Anything (SDx). SDx follows the approach that solely the software is decisive for the configuration of system functionality [23]. The concept of SDM enables a separation of the manufacturing ecosystem into physical manufacturing layers and software definition layers, which enables full scalability of production and its equipment through definition via software. Based on the description of the product to be manufactured, the whole production software, including machine control software, embedded software, cloud services, and part programs, can be automatically generated, instantiated, and configured. Key aspects for SDM are Model-Based Systems Engineering (MBSE) in production environments and digital twins of the cyber-physical systems, available at all times during the engineering process and operation [23, 36].

SDM requires a new approach to reliability analysis. Traditional production systems are characterized by fixed production lines and infrequent software updates. Figure 1 a) shows such a traditional production system. Each production system is designed and programmed for a certain task. Therefore, the reliability assessment is performed once before the operation, mostly manually, and is based on classical reliability models, such as event trees [5], fault trees [31], or reliability block diagrams [4]. In comparison to traditional production systems, the focus of modern, complex production systems is increasingly shifted towards the software part. SDM and digital twins [22] are part of this trend. SDM systems assume more frequent software updates. Depending on the domain, we can expect a few updates per day. Each update can change the system behavior significantly. Such an SDM system is illustrated in Figure 1 b). The reliability assessment of SDM systems must be performed before each software update. Therefore, it is required to conduct the reliability assessment in an automated and continuous approach. Advanced hybrid and highly flexible reliability models are the key enabling technology. These models must be automatically generated and synchronized with the available system models and digital twins. Therefore, Model-to-Model (M2M) transformation methods are another enabling technology.

This paper presents a case study on automated and continuous reliability assessment of SDM systems based on digital twins. The case study on a robotic manipulator demonstrates the applicability of the new method for automated and continuous reliability assessment of SDM based on digital twins. In particular, the method includes (i) a language profile of SysML v2 for reliability analysis, (ii) automated M2M transformations from the extended SysML v2 system models to hybrid reliability models, and (iii) probabilistic reliability analysis with the OpenPRA framework, extended with new solvers.

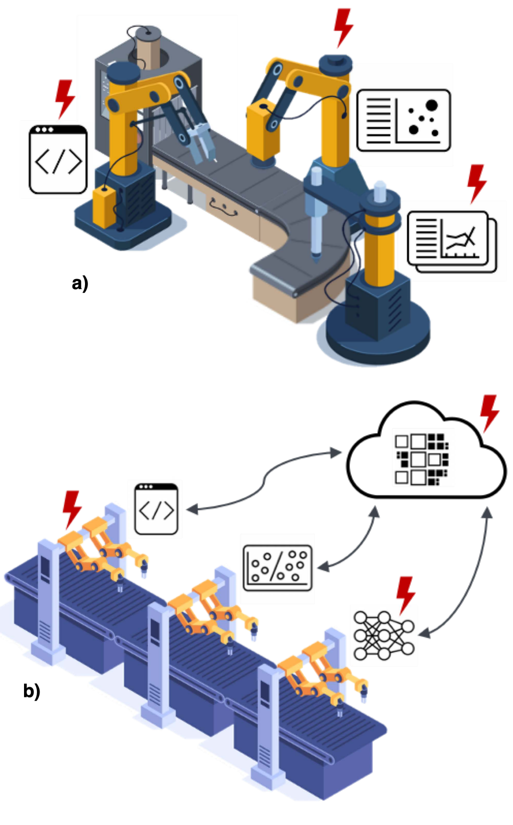

Figure 1: Comparison of a traditional production system (a), with fixed production elements and a software-defined manufacturing system (b) with flexible and re-configurable elements and a digital twin.

2 STATE OF THE ART

This section introduces the terms digital twin, model-based systems engineering, and reliability assessment. It discusses related work that is relevant to the topics examined in this paper.

2.1 Digital Twins

A digital twin is a software system consisting of models, data, and services to interact with a cyber-physical system for a particular purpose [3, 21, 22]. They serve to monitor, better understand and optimize the behavior of their respective counterparts. Modifications to the counterpart are automatically reflected in the digital twin and modifications to the digital twin are automatically reflected in the counterpart. Therefore, digital twins must provide both a representation of their counterpart and a connection to it that enables to communicate changes between both systems [9]. Digital twins enable a wide range of value adding services, such as predictive maintenance, real-time monitoring, detailed design-space exploration, process optimization, reliability assessment, and anomaly detection [29, 33, 37].

2.2 Model-Based Systems Engineering

MBSE is the formalized system modeling to support requirements, analysis, design, validation, and verification phases of the system development life cycle [17]. In this paper, we selected SysML version 2 (SysML v2) as the modeling language. We follow a simplified modeling method based on structural and behavioral diagrams.

SysML v1.X [15, 26] is a standardized modeling language. It provides system engineers with the capability to design and visualize models for various aspects of hardware and software systems and their components. SysML v2 improves the expressiveness, precision, interoperability, integration, and consistency of the language as compared to SysML v1.X. SysML v1.X was a profile of Unified Modeling Language (UML). SysML v2 is an extension of the kernel metamodel defined by the Kernel Modeling Language [KerML]. SysML v2 provides both graphical and complete textual notation [26]. Huckaby and Christensen [16] have shown, that SysML is a feasible option for modeling robotic systems. Makarov et al. [24] propose to use SysML as a modeling language to describe the digital twin. A possibility to model digital twins of cyber-physical systems is presented in Bibow et al. [3]. The applied reference architecture is specified in MontiArc.

2.3 Reliability Assessment

A Fault Tree (FT) [31] is a directed acyclic graph that models how failures can cause a system failure. In a FT, the top event is the event of interest and represents the failure of the system or subsystem. The leaves of a FT are called basic events that model the failures of individual system components. In addition to basic events, intermediate events are represented as logical gates such as AND, OR, and K/N. Logical gates show how failures in individual components can propagate through the system to a system failure. An example of a FT is shown in Figure 7. The Fault Tree Analysis (FTA) is one of the most common techniques for reliability evaluation. In this paper, we use a bottom-up [34] technique to compute the probability of system failure.

A Markov chain [11, 13] is a mathematical abstraction of a stochastic process, which consists of a set of states and transitions between them. The transition probability describes how likely it is to jump from one state to the next one and depends only upon the current state. Figure 6 gives an example of a Markov chain. The transition matrix summarizes all the transition probabilities. The most common types of Markov chains are Discrete-Time Markov chains (DTMC) and Continuous-Time Markov chains (CTMC). A DTMC describes a Markov process as a directed graph weighted with probabilities. In reliability analysis, stochastically-defined reliability-related events such as activation of faults, propagation of errors, component replacement, and repairs are considered in the system operation process described by a Markov chain.

For reliability analysis, absorbing Markov chains are the most interesting ones. A Markov chain is an absorbing Markov chain if at least one absorbing state exists and if it is possible to reach the absorbing state from every state. An absorbing state is a state which is impossible to leave, the other states are called transient states. In Figure 6 the done and failure state are absorbing states. The reliability-related questions are (a) "What is the probability that the process ends in an absorbing state?" and (b) "How many steps will it take on average?" The answers are given by the computation of the probability of absorption and the time to absorption. To quantify a Markov chain, a variety of numerical methods exist.

Hybrid reliability models: In risk analysis, different reliability models assess the reliability of the system from various points of view. This makes it important to combine different reliability models to so-called hybrid reliability models. The common way of combining fault trees, event trees, and Bayesian networks is defined as Hybrid Casual Logic [35]. The classical approach to integrate event trees and fault trees is to link fault trees to the nodes of an event tree. The quantitative analysis of such a hybrid reliability model can be performed by top-down or bottom-up techniques, for a static coherent system without common events. A method based on binary decision diagrams is suitable for non-coherent systems. In our previous work [12], we presented the limited scope of our open-source software platform OpenPRA, which supports combined event tree and fault tree analysis. We have extended it with a DTMC solver and combined Markov chain and fault tree analysis. The standard approach to link a Markov chain and fault trees is to define the transition probabilities to jump from one state to another by the failure probabilities of fault trees.

Model-to-Model transformation: The continuous reliability assessment of SDM systems requires M2M transformation methods, to create hybrid reliability models from the digital twin formalism. Transformation methods to fault trees from SysML models are presented in [25, 38], from UML in [40], from AADL in [10, 20, 32], and from Simulink in [28]. Several transformation approaches to DTMC from SysML models are introduced in [1, 6, 18, 27], from AADL in [8, 39], and from Simulink [2, 19]. None of them support the transformation to hybrid reliability models.

3 APPROACH

3.1 Overview

Our approach to the automated and continuous reliability assessment of SDM systems is shown in Figure 2. The information about the structure and behavior of the system is provided by the formalism of a digital twin. We use this information and M2M transformation methods to generate hybrid reliability models - Markov chains with interconnected fault trees. The Markov chains are created from the behavioral diagrams and the fault trees from the structural diagrams of the digital twin formalism. The hybrid reliability models are stored in the OpenPRA model exchange format and serve as input for our OpenPRA framework that can numerically solve these models. Before each update of the digital twin, the M2M transformation and the reliability assessment are repeated. The reliability assessment results are returned to the digital twin for the update/not-update decisions.

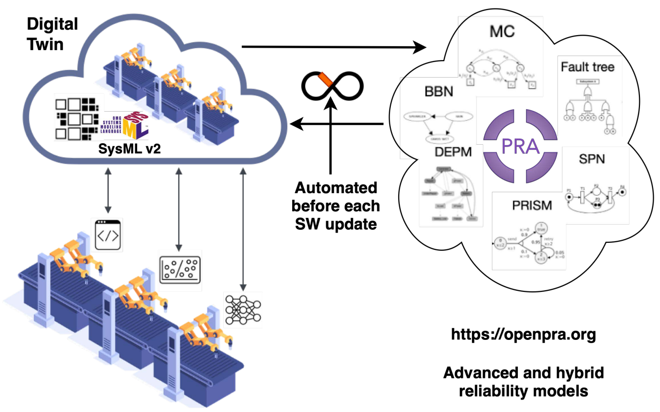
Digital Twin
SysML v2
Automated before each SW update
MC
Fault tree
BBN
DEPM
PRA
SPN
PRISM
https://openpra.org
Advanced and hybrid reliability models

Figure 2: Overview of the proposed approach to the automated and continuous reliability assessment of SDM based on digital twins.

3.2 Extended System Model

A Digital Twin models the system behavior, structure, and environment. For reliability analysis, it is also important to extend these models with reliability data. SysML v2 provides various concepts and methods to model systems, their components, behavior, structure, and environment. We assume that the DT of the SDM system is modeled with SysML v2 formalism with additional risk metadata. In SysML v2, packages and parts are used to model the structure of a system. A package is a container that organizes other elements of the model. A part models a modular structural unit, such as a system, or a component that can interact with the system either indirectly or directly. Each part can contain features such as attributes, actions, or ports. Actions model the behavior of a system and can be associated with several parts of the system structure. The actions are connected to the sequencing of actions that represent the software flow of the system. This is illustrated in Figure 6. The sequencing of actions can be controlled by control nodes such as decision node, fork node, join node, and merge node. The RiskMetadata package, provided by SysML v2, allows us to embed reliability data, such as failure probabilities to parts or actions. We extended the RiskMetadata package with the possibility to mark redundant components. In this paper, we consider only the classical redundancy. However, in a similar way, we can model other reliability-related features such as k/n switches, spares, dependencies for dynamic fault trees, etc.

3.3 M2M Transformation Method

Our M2M transformation algorithm parses a SysML v2 file and searches for keywords. Which allows us to read the structure, dependencies, behavior, and reliability data for further processing.

A function creates the XML structure according to the OpenPRA model exchange format and writes the final XML file. Figure 3

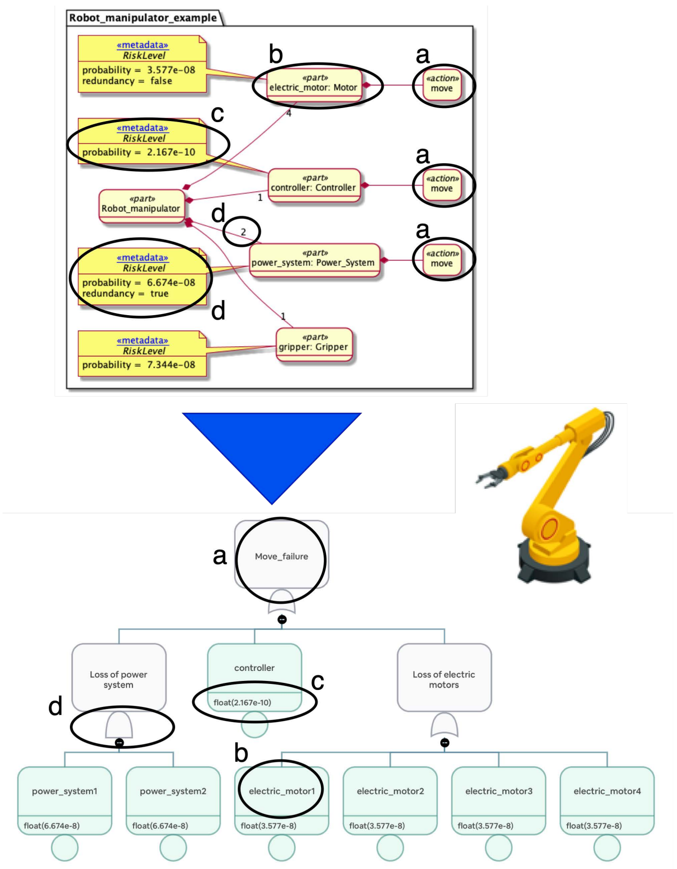

Figure 3: M2M transformation from SysML v2 structure models to fault trees.

shows the M2M transformation method for fault trees. Our algorithm automatically generates an FT for each action, where only the used parts appear as basic events. For example, the action *move* (a) is transformed into the top event modeled by an OR gate. Our algorithm transforms the parts associated with the *move* action, such as the *electric_motor* (b), into basic events. The failure probabilities for each basic event (c) are provided by the risk metadata package. In addition, the risk metadata gives us information about the redundancy. The redundancy of the *power system* is modeled by an AND gate (d).

Figure 4 illustrates the M2M transformation method for Markov chains. The actions (*move, pick, move_with_piece, and release*) are transformed into transient states and one or several absorbing failure states are created. The fault trees provide the transition probabilities. The transition probability to jump from the *move* state to the *failure* state is given by the *Move_failure* FT from Figure 3. The transition probability to jump from *move* to *pick* is $1 - P(Move\_failure)$. The generated hybrid reliability models in the OpenPRA model exchange format contain a Markov chain with interconnected fault trees.

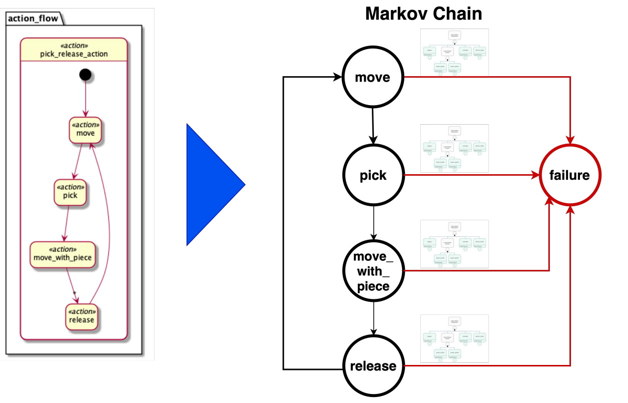

Figure 4: M2M transformation from SysML v2 action sequences to Markov chains.

3.4 Reliability Analysis

We use and extend our OpenPRA framework [12] for the reliability analysis. OpenPRA is an open-source framework, which aims to integrate multiple Probabilistic Risk Assessment (PRA) methods into a universal, easy-to-use, and highly customizable environment. OpenPRA includes a FTA module, a new DTMC module, and a new integrated analysis module. The FTA module consists of a public API, a solver, and an XML reader. The FT solver is based on a bottom-up algorithm that computes the probability of failure of the top event. In the scope of this work, we extended OpenPRA with a DTMC module and an integrated analysis module for hybrid reliability models. The DTMC module consists of similar modules as the FTA module. The DTMC solver can compute the probability of absorption and the time to absorption. The integrated analysis module was developed to analyze hybrid reliability models, such as Markov chains linked with fault trees.

The integrated analysis module implements the following steps: (i) Reads the given XML files and stores the reliability models internally as graphs. (ii) Start solving the fault trees by calling the FTA solver, since they are at the bottom level and linked to the DTMC. (iii) Add the computed results to the transitions of the Markov chain. (iv) Call the DTMC solver to solve the final DTMC. (v) Output the final results of each fault tree, the probability of absorption, and the time to absorption of the DTMC.

4 CASE STUDY

4.1 System Overview

For our case study we have selected a robotic manipulator. The robotic manipulator consists of a control system, a camera, seven non-redundant electric motors, seven non-redundant torque sensors, two redundant power systems, and a two-finger gripper. The initial software to control the robot with the two-finger gripper, Figure 5 a), consists of the following steps: initially, the robotic manipulator moves to a certain position and subsequently tries to detect a workpiece to pick. If the piece is detected, the robotic manipulator picks the piece, moves to the conveyor belt, and releases the piece there. Afterward, the robotic manipulator starts the process again. In case there is no detected workpiece, the task of the robotic manipulator is finished.

It is possible to install new tools on the robotic manipulator. For example a soldering iron. This leads to the necessity to update the control software. The updated control software, Figure 5 b), consists of the following steps: initially, the robotic manipulator tries to detect a solder spot on a board. If there is a detected spot, the robotic manipulator moves to this position. After moving to the detected position, the position is checked one more time. If the position is correct, the robotic manipulator starts the soldering. If not, the adjust action is performed. After the soldering is completed, the robotic manipulator tries to detect another solder spot. If no spot is detected, the task is complete.

It is possible to update the software by adding new functions or skills, e.g. adding a function to rotate the workpiece into the pick and release software flow (Figure 5 a)).

4.2 System Models

The system is modeled in SysML v2 with a high-level package and parts. Figure 8 illustrates the system architecture model of the system. The components of the robotic manipulator (a) are modeled as parts and extended with risk metadata. The risk metadata adds the failure probability and if necessary redundancy to the component, e.g. part *power system* with failure probability and redundancy (d). The failure probability is obtained from the FIDES 2009 [14] and NPRD-95 [7]. The actions are connected to the corresponding parts. For example, the action *detect* is added to the parts *camera*, *power system*, and *control system* (c). In SysML v2 it is possible to add risk metadata to actions. We use this for instance to override the failure probability of the *electric motor* during the *move_with_piece_action* with a higher failure probability (b), due to the additional weight.

The behavior of the robotic manipulator, which shows the software flow, is modeled as a sequence of actions. Figure 5 a) and b) depict the software flows modeled in SysML v2.

4.3 Hybrid Reliability Models

The software flow, 5 a), is automatically transformed into a Markov chains with interconnected fault trees. Figure 6 illustrates the Markov chain of the pick and release action. The Markov chain contains the transient states *move*, *detect*, *pick*, *move with a piece*, and *release* as well as the absorbing states *done* and *failure*. After each transient state, it is possible to jump to the *failure* state. These transition probabilities are given by the corresponding fault trees. Similarly, we automatically transform every updated software flow into a Markov chain with interconnected fault trees. The Markov chain of the soldering iron control software contains the transient states *detect*, *move*, *check position*, *adjust*, and *solder* as well as one absorbing failure state for each action and the absorbing state *done*.

From the system architecture, as shown in Figure 8, our M2M transformation method creates automatically for each action a fault tree containing all the used components. The probabilities of basic events define the failure probabilities of associated components during one minute of operation. The data is obtained from the FIDES 2009 [14] and NPRD-95 [7]. Figure 7 illustrates the fault tree of the detect action. The *detect_failure* is modeled as the top event. The top event occurs either because of the failure of the *camera*, *control system*, or *power system*. The power system fails if both *power systems* fail. The other fault trees are not shown in the scope of this paper.

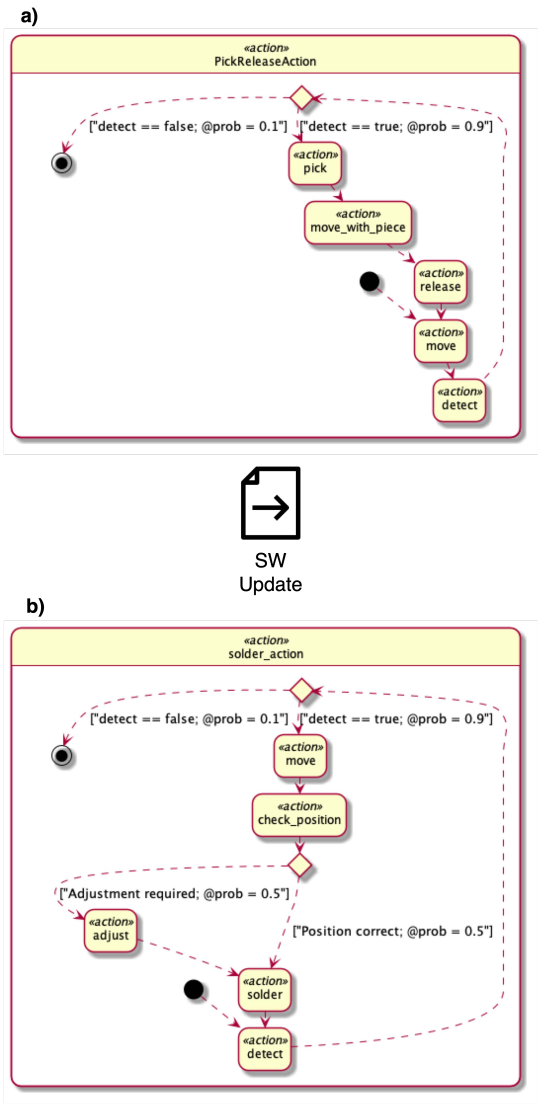

Figure 5: The software flow of the pick and release action (a) and the solder action (b).

5 RESULTS

Table 1: Probability of absorption of the Markov chains. The probability of absorption is given for one absorbing failure state.

Markov chain	Probability of absorption
Pick and release	1.690e-05
Soldering	7.802e-06

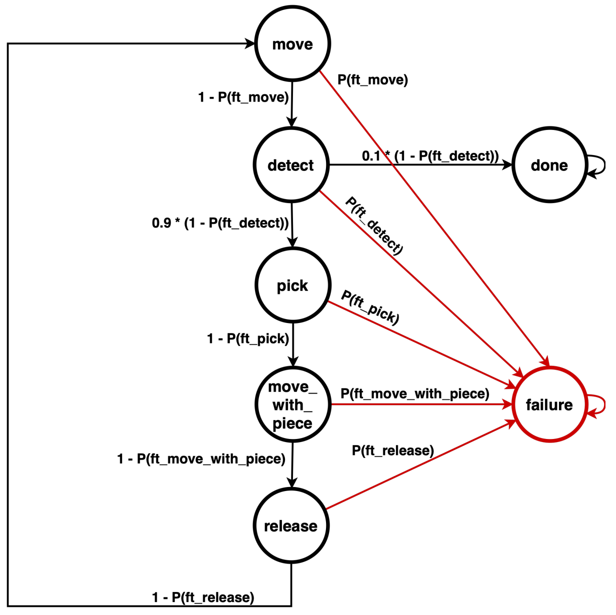

Figure 6: Markov chain of the pick and release action. With two absorbing states failure and done. From each transient , it is possible to jump directly to the failure state.

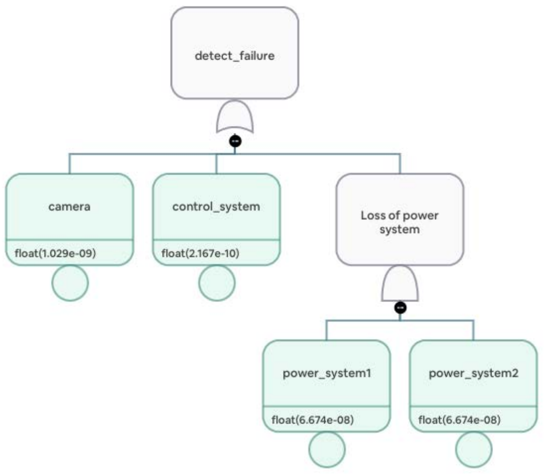

Figure 7: Fault tree of the detect failure. The top event detect failure occurs either because of the failure of the camera, control system or power system.

We analyze the reliability of the hybrid reliability models, discussed in subsection 4.3, with the OpenPRA framework. The obtained results are presented in this section. First, we consider the hybrid reliability model of the pick and release action. Second, we focus on the soldering action. Table 1 presents the results of the Markov chains with one failure state. The starting state in the pick and release Markov chain is the *move* state and in the soldering Markov chain the *detect* state. The first row of Table 1 shows that the probability of absorption in the failure state of the pick and release action is 1.690e-05. The probability of absorption in the failure state of the soldering action, shown in the second row of Table 1, is 7.802e-06. Table 2 shows the probabilities of absorption of the Markov chain of the pick and release action. Starting in the *move* state and with several absorbing failure states. The probability to get absorbed in the *move failure* or *release failure* state is most likely. In comparison Table 3 shows the results of the Markov chain of the solder action, starting in the *detect* state, with several absorbing failure states. The *adjust failure* and *move failure* states are the most likely absorbing states. For completeness, Table 4 presents the top event failure probabilities of all given fault trees.

Table 2: Probability of absorption of the Markov chain of the pick and release action, with a failure state for each action.

Failure state	Probability of absorption
move failure	6.648-06
detect failure	1.246e-08
pick failure	6.722e-07
move with piece failure	8.237e-06
release failure	1.332e-06

Table 3: Probability of absorption of the Markov chain of the solder action, with a failure state for each action.

Failure state	Probability of absorption
detect failure	1.246e-08
move failure	5.983e-06
check position failure	1.121e-08
adjust failure	1.132e-06
solder failure	6.629e-07

Table 4: Failure probabilities of the fault trees.

Fault tree	Failure probability top event
Move	6.647e-07
Detect	1.246e-09
Pick	7.468e-08
Move with piece	9.152e-07
Release	1.480e-07
Check position	1.246e-09
Adjust	2.516e-07
Solder	7.366e-08

The results show, that our method is suitable for the automated and continuous reliability assessment of SDM systems based on digital twins. The structure and behavior of the system are modeled in a SysML v2 formalism extended for reliability analysis. We can compute the reliability of the system depending on the software. The results of the fault trees are checked and compared against XFTA [30].

6 CONCLUSION

In this paper, we use a case study on a robotic manipulator to show, that our method for automated and continuous reliability assessment of SDM systems with frequently changing software is suitable. The robotic manipulator can perform different tasks depending on the uploaded software. The digital twin of the robotic manipulator is modeled in SysML v2. Our method consists of (i) an extension of SysML v2 for reliability analysis, (ii) automated M2M transformations from the extended SysML v2 system models to hybrid reliability models, and (iii) probabilistic reliability analysis with our developed OpenPRA framework extended with new solvers. The case study demonstrates, that the automatically generated hybrid reliability models can adapt to changes in system structure and behavior. This enables the possibility to compute the reliability of the system before each software update, based on the models of the digital twin. The main goal of the paper is to demonstrate how the automated and continuous reliability assessment can enable the utilization of safety-critical SDM systems.

The extension to other hybrid reliability models that include stochastic Petri net, dual graph error propagation model, CTMCs, or dynamic fault trees is possible. This will require the further extension of the SysML v2 formalism, the development of new solvers for our OpenPRA framework, and new M2M transformations. This will help us to analyze the reliability of the system more precisely and will be the subject of our future work. It is possible to extend the OpenPRA framework to other reliability metrics, such as MTTF, Weibull distribution, or exponential distribution.

The M2M transformation algorithms are limited in their functionality. The transformation from behavioral diagrams to Markov chains does only support decision nodes, no join nodes, no fork nodes, and no merge nodes. The transformation algorithm to fault trees does not support special features, such as ports or attributes.

ACKNOWLEDGMENTS

This work has been partly funded by the German Federal Ministry of Economic Affairs and Climate Action (Bundesministerium für Wirtschaft und Klimaschutz, BMWK) under the project "Software-defined Manufacturing für die Fahrzeug- und Zulieferindustrie" (SDM4FZI, funding code 13IK001ZE).

REFERENCES

[1] Abdelhakim Baouya, Djamal Bennouar, Otmane Ait Mohamed, and Samir Ouchani. 2015. A probabilistic and timed verification approach of SysML state machine diagram. In *2015 12th International Symposium on Programming and Systems (ISPS)*. IEEE, 1–9.
[2] Adrian Beer, Todor Georgiev, Florian Leitner-Fischer, and Stefan Leue. 2013. Model-based quantitative safety analysis of Matlab Simulink/Stateflow models. In *Model-Based Development of Embedded Systems*. 60–69.
[3] Pascal Bibow, Manuela Dalibor, Christian Hopmann, Ben Mainz, Bernhard Rumpe, David Schmalzing, Mauritius Schmitz, and Andreas Wortmann. 2020. Model-driven development of a digital twin for injection molding. In *International Conference on Advanced Information Systems Engineering*. Springer, 85–100.
[4] Marko Čepin. 2011. Reliability block diagram. In *Assessment of Power System Reliability*. Springer, 119–123.
[5] PL Clemens. 2002. Event tree analysis. *JE Jacobs Sverdrup*, (2002).
[6] Mourad Debbabi, Fawzi Hassaine, Yosr Jarraya, Andrei Soeanu, and Luay Alawneh. 2010. *Verification and validation in systems engineering: assessing UML/SysML design models*. Springer Science & Business Media.
[7] William Denson, Greg Chandler, William Crowell, Amy Clark, and Paul Jaworski. 1994. *Nonelectronic parts reliability data 1995*. Technical Report. RELIABILITY ANALYSIS CENTER GRIFFISS AFB NY.
[8] Yun-wei Dong, Geng Wang, and Hong-bing Zhao. 2009. A model-based testing for aadl model of embedded software. In *2009 Ninth International Conference on Quality Software*. IEEE, 185–190.
[9] Romina Eramo, Francis Bordeleau, Benoit Combemale, Mark van Den Brand, Manuel Wimmer, and Andreas Wortmann. 2021. Conceptualizing digital twins. *IEEE Software* 39, 2 (2021), 39–46.
[10] Peter Feiler and Julien Delange. 2017. Automated fault tree analysis from aadl models. *ACM SIGAda Ada Letters* 36, 2 (2017), 39–46.
[11] Norman B Fuqua. 2003. The applicability of markov analysis methods to reliability, maintainability, and safety. *Selected Topic in Assurance Related Technologies (START)* 2, 10 (2003), 1–8.
[12] Philipp Grimmeisen, Artur Karimov, Mihai A Diaconeasa, and Andrey Morozov. 2021. Demonstration of a Limited Scope Probabilistic Risk Assessment for Autonomous Warehouse Robots With OpenPRA. In *ASME International Mechanical Engineering Congress and Exposition*, Vol. 85697. American Society of Mechanical Engineers, V013T14A030.
[13] Charles Miller Grinstead and James Laurie Snell. 1997. *Introduction to probability*. American Mathematical Soc.
[14] FIDES Group. 2010. *FIDES guide 2009, Reliability Methodology for Electronic Systems*.
[15] Matthew Hause et al. 2006. The SysML modelling language. In *Fifteenth European Systems Engineering Conference*, Vol. 9. 1–12.
[16] Jacob Huckaby and Henrik I Christensen. 2014. A case for SysML in robotics. In *2014 IEEE International Conference on Automation Science and Engineering (CASE)*. IEEE, 333–338.
[17] SE INCOSE. 2007. Vision 2020 (INCOSE-TP-2004-004-02).
[18] Yosr Jarraya, Andrei Soeanu, Mourad Debbabi, and Fawzi Hassaine. 2007. Automatic verification and performance analysis of time-constrained sysml activity diagrams. In *14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'07)*. IEEE, 515–522.
[19] Anjali Joshi and Mats PE Heimdahl. 2005. Model-based safety analysis of simulink models using SCADE design verifier. In *International conference on computer safety, reliability, and security*. Springer, 122–135.
[20] Anjali Joshi, Steve Vestal, and Pam Binns. 2007. Automatic generation of static fault trees from AADL models. (2007).
[21] Jörg Christian Kirchhof, Judith Michael, Bernhard Rumpe, Simon Varga, and Andreas Wortmann. 2020. Model-driven digital twin construction: synthesizing the integration of cyber-physical systems with their information systems. In *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*. 90–101.
[22] Werner Kritzinger, Matthias Karner, Georg Traar, Jan Henjes, and Wilfried Sihn. 2018. Digital Twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine* 51, 11 (2018), 1016–1022.
[23] Armin Lechler, Oliver Riedel, and Daniel Coupek. 2017. VIRTUAL REPRESENTATION OF PHYSICAL OBJECTS FOR SOFTWARE DEFINED MANUFACTURING. *International Conference on Production Research* (2017).
[24] Makarov, Frolov, Parshina, and Ushakova. 2019. The design concept of digital twin. In *2019 Twelfth International Conference" Management of large-scale system development"(MLSD)*. IEEE, 1–4.
[25] Faida Mhenni, Nga Nguyen, and Jean-Yves Choley. 2014. Automatic fault tree generation from SysML system models. In *2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. IEEE, 715–720.
[26] Object Management Group (OMG). 2021. *OMG Systems Modeling Language (SysML) Version 2.0*. See also URL https://github.com/Systems-Modeling/SysML-v2-Release..
[27] Samir Ouchani, Otmane Ait Mohamed, and Mourad Debbabi. 2014. A formal verification framework for SysML activity diagrams. *Expert Systems with Applications* 41, 6 (2014), 2713–2728.
[28] Yiannis Papadopoulos and Matthias Maruhn. 2001. Model-based synthesis of fault trees from matlab-simulink models. In *2001 International Conference on Dependable Systems and Networks*. IEEE, 77–82.
[29] Jerome Pfeiffer, Daniel Lehner, Andreas Wortmann, and Manuel Wimmer. 2022. Modeling Capabilities of Digital Twin Platforms-Old Wine in New Bottles? (2022).
[30] Antoine Rauzy. 2020. *Probabilistic Safety Analysis with XFTA*.
[31] Enno Ruijters and Mariëlle Stoelinga. 2015. Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools. *Computer science review* 15 (2015), 29–62.
[32] Hongyu Sun, Miriam Hauptman, and Robyn Lutz. 2007. Integrating product-line fault tree analysis into aadl models. In *10th IEEE High Assurance Systems Engineering Symposium (HASE'07)*. IEEE, 15–22.
[33] Fei Tao, Qinglin Qi, Lihui Wang, and AYC Nee. 2019. Digital twins and cyber–physical systems toward smart manufacturing and industry 4.0: Correlation and comparison. *Engineering* 5, 4 (2019), 653–661.
[34] William E Vesely, Francine F Goldberg, Norman H Roberts, and David F Haasl. 1981. *Fault tree handbook*. Technical Report. Nuclear Regulatory Commission Washington DC.

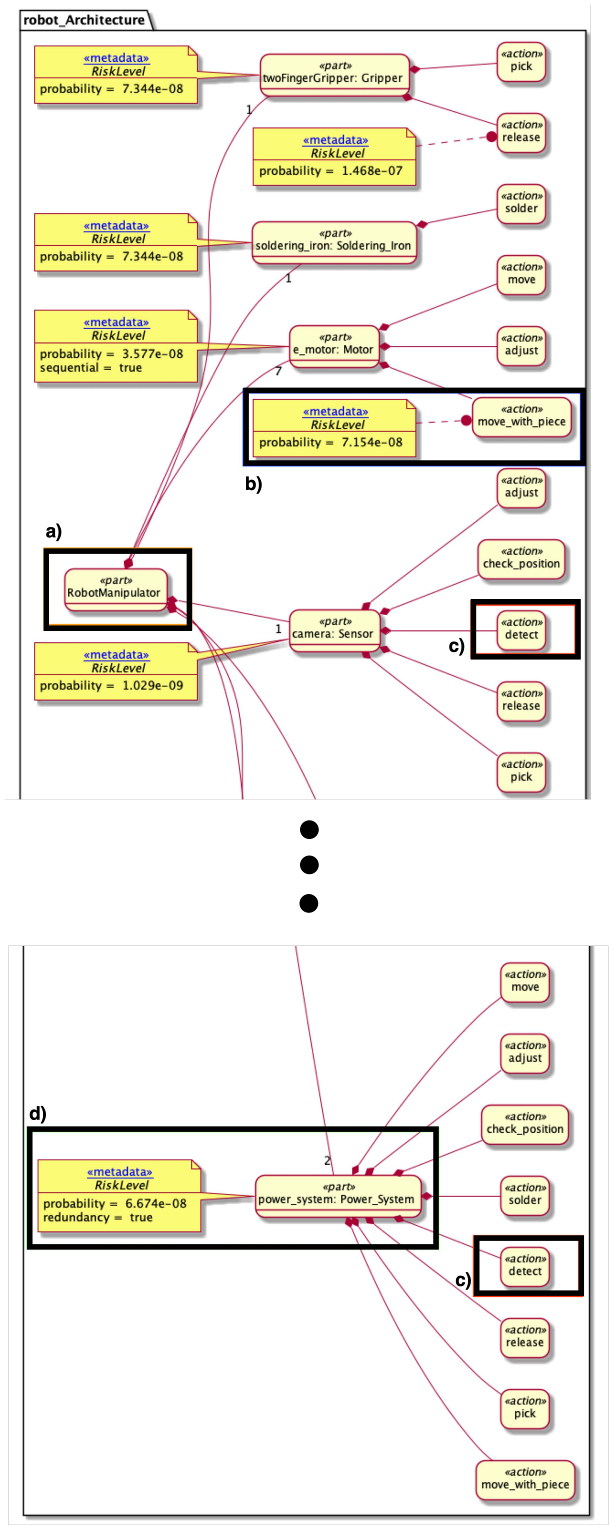

Figure 8: Except of the system structure of the robotic manipulator modeled in SysML v2.

[35] Chengdong Wang. 2007. *Hybrid causal logic methodology for risk assessment.* University of Maryland, College Park.

[36] Lei Xu, Lin Chen, Zhimin Gao, Hiram Moya, and Weidong Shi. 2021. Reshaping the Landscape of the Future: Software-Defined Manufacturing. *Computer* 54, 7 (2021), 27–36.

[37] Qinghua Xu, Shaukat Ali, and Tao Yue. 2021. Digital twin-based anomaly detection in cyber-physical systems. In *2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST).* IEEE, 205–216.

[38] Nataliya Yakymets, Hadi Jaber, and Agnes Lanusse. 2013. Model-based system engineering for fault tree generation and analysis. In *International Conference on Model-Driven Engineering and Software Development*, Vol. 2. SCITEPRESS, 210–214.

[39] Tao Zhang, Yechun Jiang, Junda Ye, Cheng Jing, and Huamin Qu. 2014. An aadl model-based safety analysis method for flight control software. In *2014 International Conference on Computational Intelligence and Communication Networks.* IEEE, 1148–1152.

[40] Zhao Zhao. 2014. *UML model to fault tree model transformation for dependability analysis.* Ph. D. Dissertation. Carleton University.