Is Model-Based Dependability with Knowledge Bases a Tool for the Future?

Ola Bäckström[a*], Pavel Krcal[b]
[a]RiskSpectrum AB, Stockholm, Sweden
[b]RiskSpectrum AB, Dresden, Germany

Abstract: Model-based approaches to dependability offer multiple advantages for both system and reliability engineers. A system model is created and analyzed on a level corresponding to the system design model. This facilitates the communication between a system engineer and a dependability analyst. Tool support allows for automated analyses with the resolution of any standard low-level dependability formalism. The system behavior is well defined by the underlying semantics pre-programmed in the tool.

Availability analyses combine system structure including all redundancies, reconfigurations, and backup strategies with failure and repair behavior of the individual components. Analysis techniques range from fault trees to full-fledged simulations, offering different trade-offs between the speed and the precision. Model-based availability assessment allows analysts to build models on a high level, choose the analysis technique and automatically generate a low-level model for the preferred analysis.

RiskSpectrum ModelBuilder/KB3 is a platform for model-based dependability analysis that must be first filled by a library of components defining their behavior and mutual interactions. These libraries are called knowledge bases. A knowledge base guides the ModelBuilder in generating low-level models for further processing. It is up to the experts defining the knowledge base to determine which types of analyses shall be possible and which features shall be included in the low-level models.

In this paper, we discuss which types of dependability analyses can be encoded in knowledge bases, how one can create models, especially possibilities given by an integration of Model Based Dependability and Model Based Systems Engineering (MBSE) tools, and what types of results can one produce for a respective analysis type. Furthermore, we propose methods for validating models, and explore possibilities of automatic search for design optima.

Keywords: Model-based dependability, Availability Analyses, Knowledge Bases, Dynamic Calculations.

1. INTRODUCTION

Model-based approaches to dependability (a summary term for reliability, availability, maintainability and safety) receive increased interest for industrial applications. It represents a step towards streamlining the design evaluation from the dependability perspective. Two aspects go hand in hand in this process. Automatization of repetitive tasks can cover larger areas when the expert input needed to perform these tasks is formally encoded in a way that can be utilized by a computer program.

The ultimate goal of a model-based dependability analysis is to take the system description in a language used by a system engineer, let a dependability engineer specify the configuration and dependability measures of interest, automatically extract a dependability model compatible with a suitable solver, and feed the results back to the system description in a form understandable by a system engineer.

The first step in this process transforms the system model into a high-level dependability model by screening out irrelevant parts and equipping the remaining ones with dependability information. This model has a value of its own. It facilitates the communication between a system engineer and a dependability analyst by sharing the same (or very similar) language describing the system. It becomes much easier to modify or adapt the studied model to, for instance, evaluate different scenarios or design alternatives.

In the next step, an analyst sets up a scenario for evaluation. This includes specifying a configuration of the system, boundary conditions, and dependability measures of interest. Performing this on a high-level increases the transparency of the analysis.

The last step generates a low-level description of the model that serves as an input to a solver. This is a separate model that can be also reviewed or, if need be, modified manually. But one of the goals of the model-based approach is to hide this modeling level from everyday dependability work.

RiskSpectrum ModelBuilder, a fully commercial version of the tool KB3 developed by EDF, is a tool for model-based dependability studies. It is a platform that must be first filled by dependability information for a specific domain. This has the form of a library of components. For each of them, we define their behavior and interactions with other components. These libraries are called knowledge bases. A knowledge base defines system behaviors which emerge from system interactions reacting to stochastic events. System behaviors then specify how the tool generates low-level models for further processing. It is up to the experts defining the knowledge base to determine which types of analyses shall be possible and which features shall be included in the low-level models.

In this paper we discuss drivers for adopting a model-based approach, starting from a fault tree generation perspective, thereafter we discuss challenges for fault trees in advanced applications and how model based approaches can help and then, finally, how verification of model generation can be achieved when using model based approaches.

2. HISTORY AND FUTURE OF FAULT TREE GENERATION

Development of fault tree models has to a large degree been performed in the same way for quite many years. The coverage in the fault trees as well as the functionality provided by the tools, for example RiskSpectrum or CAFTA, has evolved over the years. Sometimes the development of the tooling has taken leaps, such as the inclusion of calculation methods like Binary Decision Diagrams to increase accuracy [1,2] or with the inclusion of conditional quantification approaches [3]. Yet, the underlaying approach remains the same.

We can mainly see two drivers for drastically changing the approach:

- Dynamic approaches
- Digital transformation

Fully dynamic approaches, like dynamic definition of event trees, have been developed over a significant period of time, see for example the overview in [4] mentioning MCDET, PyCATSHOO, RAVEN, ADAPT etc. With fully dynamic approaches we in this context refer to codes that will interact with Thermal hydraulic analysis to define success criteria automatically. The approaches are definitively interesting on their own merits, but considering the time it takes to solve the static PSA models – with the existing computer power, such approaches will not be able to replace the need of static PSA tools.

Dynamic approaches will surely play a very important role in the definition of sequences, and assessment of interactions that can hardly be analyzed with static PSA tools – like passive system failure probabilities.

A certain class of dynamic approaches focuses on the stochastic behavior of discrete event systems, possibly in continuous time. As opposed to fully dynamic approaches, the deterministic part (modeling of physical phenomena) is limited to relatively simple arithmetic calculations. The system is event-driven, where events occur at random time points determined by associated distributions. After a new event occurs, the system reacts to it by changing its state. When the new state is reached, the system waits for the new event to occur.

A central notion in behavior of these systems is the state of the system and its evolution in time. Failure behaviors are sets of runs of the system reaching a failed state. These sets of runs can be characterized by finite sets of transitions between system states. An analysis of such a system must identify failure behaviors and quantify them.

These types of dynamic models cannot replace static Probabilistic Safety Assessment (PSA) models and tools either. Analysis of these dynamic models does not scale to the size and resolution of PSA models, at least those used in nuclear safety. Another advantage of static PSA models is the exhaustiveness of the analysis, covering all (relevant) combinations of failures. Also, the type of results obtained from static PSA

is relatively easy to understand and review, which makes it suitable for risk-informed decisions. Yet, dynamic models can express system features that can be only roughly approximated by static fault trees. This makes them attractive to analysts when the precision of the result depends on including such features. For certain reliability or availability studies, one needs to resort to dynamic models to obtain meaningful results.

What do we mean by digital transformation? According to IBM web page on digital transformation [5] *Digital transformation is a strategic initiative that incorporates digital technology across all areas of an organization. It evaluates and modernizes an organization's processes, products, operations and technology stack to enable continual, rapid, customer-driven innovation.*

Looking at a process to design a PSA model, it can be illustrated by the image below

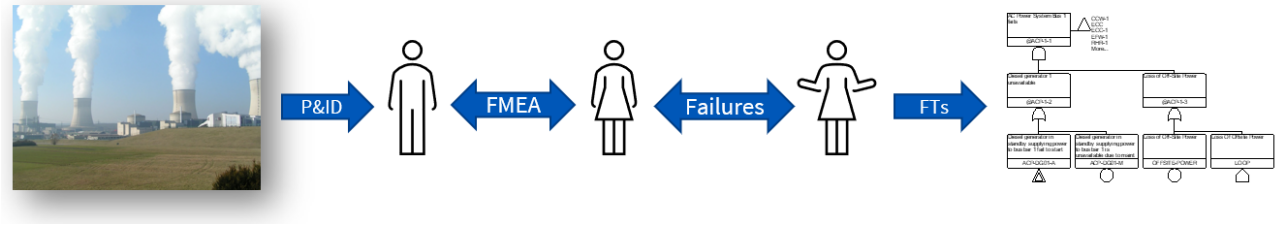

Figure 1. An example of the process to perform a systems analysis in a PSA. Piping and Instrumentation Diagrams (P&IDs) are processed by a Failure Mode and Effects Analysis (FMEA), and together with the structure of failure propagation through the system, give rise to a model based on Fault Trees.

Considering the amount of manual work included in the process, it is clear that we are far from digital transformation in the PSA community. Is it desirable to expose ourselves and our habits to digital transformation? We would say that it is. The reason is that achieving the essential goals of PSA requires constant attention to quality and efficiency. Both can be improved by adopting model-based approaches in the PSA process.

Can Artificial Intelligence be used in the process of defining a PSA model? This is surely a question that will be debated a lot over the years to come. A main question in using AI in the design of a risk assessment is the confidence we can expect from the answer. Generally, the confidence that we expect from the models of risk assessment needs to be very high. And we need a very high confidence level in the representation of machines/systems that shall have a low (very low) failure probability. Therefore, a PSA model is likely not, within a near future, a candidate for a black box.

But do we actually need AI for digital transformation within PSA? The answer is no. Model Based Systems Engineering (MBSE) and the corresponding Model Based Safety Assessment (MBSA) will provide the means for digital transformation. In a simplified form, the process illustrated in figure 1 can be changed to an automated process, in this case starting from a set of P&IDs (design flow charts, preferably from Computer Aided Design (CAD) software). But the starting point could equally be any MBSE model.

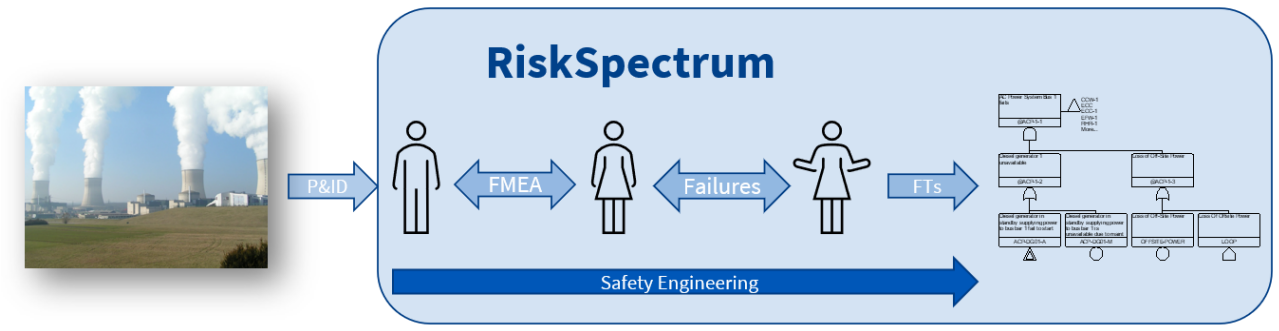

Figure 2. A digital transformation of the setup of fault trees, in this case from P&IDs

The handcraft of building fault trees from P&IDs is actually ideal to be replaced by digital transformation. This conditional that you have a set of well-defined rules for transforming P&IDs into fault trees. In essence, this is what is defined in the MBSA tooling. For RiskSpectrum ModelBuilder, these rules are referred to as a knowledge base (KB). Integration with different types of systems for MBSE or CAD is an ongoing project by EDF and RiskSpectrum.

The quality assurance process may also be significantly improved, as suggested by Dirksen [6,7], where in its implementation the failure modes generated by the system can be automatically checked versus the logic of the P&IDs.

The obvious question is: Given that the fault trees can be automatically generated, why has this not yet been adopted by all users already? There has to be some hidden issue.

A main issue in the adoption of MBSA approaches is the existence of the current model. Which has been constructed without the use of MBSA. Is this a good argument for not using MBSA approaches? Not really, as you can start building out the model for new or modified systems. An important feature of the MBSA tool is that not everything has to be defined by the tool – you still need the flexibility to add features.

Would it be possible to reverse engineer the fault trees back into P&IDs. This would likely be a jump start of the use of MBSA for existing models. Initial research in this direction has been reported in [8]. It appears not to be completely straightforward, as the modelling techniques of different users can differ. Which again is a good reason for letting a "robot" do this work.

The main reason for the still low adoption of these approaches appears to be habits and fear. Short deadlines, with incremental addition of features to a model is likely the enemy. The digital transformation in the PSA industry will come when the management adapts a digital transformation strategy and starts questioning the labor intense work being put into the models. The tools are available.

3. SOME EXAMPLES WHERE MODEL-BASED ASSESSMENT IS SUPERIOR TO STATIC FT

Despite the advances that have been made over the years regarding static fault tree tools, there are still numerous limitations to cover a wider range of applications. Put in other words, the existing tools perform a rather defined task which can be constructed in the form of a static fault tree.

There are a number of different tasks where the static fault tree will not be sufficient on its own, and you will need a tool or framework that is more flexible. In this paper we discuss following:

- Integration with other plant systems
- In depth assessment where more dynamic features or behaviors are needed
- Availability assessments where simulation capabilities are wanted

3.1. Integration with other plant systems

Plant integration, typically as part of a digital transformation strategy, cannot easily be done directly with fault trees. To some extent, the integration can be done directly with the PSA tool with regard to for example

- Different types of data, for example;
    - Reliability data generated directly from the plant data
    - Initiating event frequencies
- Results output, for example
    - Top results
    - Importance results

Integration of data can be relevant for an average PSA model; a plant database for collecting information about the components and thereby be able to generate plant specific reliability data using for example Bayesian approaches. Similarly for initiating event frequencies. Integration of data could also be done in the context of risk- or trip monitors. In this context the integration is to read maintenance plans (for example outage) to be able to evaluate such in the risk monitor. Or to track the online risk by integrating with the

plant information system (event log). Such integration is rather straight forward and is reasonably easy to achieve.

But if a more comprehensive integration is desired, like a direct link between the P&IDs and the fault tree, or from the FMEA directly to the fault tree a tool for MBSA would be desirable. This is covered in the previous sections and hence not further discussed here. A vision would be that fault tree model is based on the real P&IDs, where a simplified P&ID is (automatically) defined based on the relevant components – which in its turn generates the fault trees on which the risk is calculated. This would mean that there is a automated and transparent process, and that the risk monitor could also be based directly on the simplified P&IDs to simplify the common understanding. This would be digital transformation in a PSA and risk monitor model context.

3.2. In depth assessment where dynamic features or behaviors are needed

There are numerous situations where a static fault tree has challenges to represent the reliability or availability of the system sufficiently. Looking at an electrical system from a PSA perspective, this is normally a part of the model that already is complex to represent sufficiently. There are for example dependencies between high power and low power systems, there are different time intervals that needs to be considered, there are situations where only batteries can be considered and there are situation where only diesels can be considered in long term loss of offsite power scenarios. However, these models are still highly simplified. They may be sufficient for the purpose of the PSA objective, but not sufficient if you are faced with the question of reliability or availability of the system (not conditional that you are looking at the failure scenario). You may for example be interested in understanding the availability of some busbars.

In these cases, you would need to consider a wider scope of the analysis. There are cascades of transitions that may be initiated by a failure and a need to reconfigure the system. These transitions should be carried out in a predefined way, the impact of a short circuit can affect both downwards and upwards, there are repairs that can – and should – be considered, there are looped interactions, and the available time represented by the batteries should be considered in an appropriate manner. A practical example, a benchmark, of such a system was introduced in [9].

An approach, also discussed within the same paper, is the use of a Markov process to represent the situation. More specifically, a method called Boolean Driven Markov Process (BDMP) [11] is suggested. Some main features of the BDMP approach are triggers (to steer the order of alternatives) and consideration of timing, which allows for consideration of repairs in a correct way. Different alternatives for evaluating BDMPs are evaluated in [10].

Modeling with BDMPs resembles building fault trees, which brings also multiple advantages, especially for analysts familiar with fault trees. On the negative side, this formalism is also very low-level, far from the system description by, e.g., single-line diagrams. It is possible to define a compact knowledge base that provides components for building single-line diagrams where one can model this power supply system with complex priority-based reconfiguration strategies. The availability analysis of this system by the means of Monte Carlo simulations explores behaviors where the system responds to various failures by reconfigurations and attempts to repair failed components. The results reduce conservatism compared to analyses based on fault trees with repairs or BDMPs [12].

The electrical system and the BDMP is only an example to illustrate a type of calculation that requires more capabilities. Another example could be state triggered situations, where there can be waiting times involved. Consider a repair dependent on states, where there are multiple states considered. The reliability and the repair time of the individual component depends on its state (as good as new, some degradation, significant degradation, failed/repaired). In this case it may be relevant to consider the use of a Petri Net. These types of approaches are fully feasible in a model based approach like RiskSpectrum ModelBuilder.

3.3. Availability assessments when simulations may be needed

Availability analyses combine system structure including all redundancies, reconfigurations, and backup strategies with failure and repair behavior of the individual components. Analysis techniques range from

fault trees to full-fledged simulations, offering different trade-offs between the speed and the precision. Model-based availability assessment allows analysts to work with models on a high level, choose the analysis technique and automatically generate a low-level model for the preferred analysis. A necessary condition for this is a knowledge base describing behaviors of components relevant to availability, possible failures and repairs and their effects on the rest of the system.

An example where model-based assessment would be more suitable to perform an availability assessment is optimization of repair strategies for digital Instrumentation and Control (I&C). The question to be answered in this case is: how many spare parts should be allocated in consideration of the system design.
Compared to a standard spare parts optimization also the system design is considered, where the failure impact can be considered.

4. EXAMPLES OF POTENTIAL ANALYSES

In this section we exemplify some types of problems where advanced dependability approaches are useful:

- Electrical distribution availability assessment
- Availability assessment of a hybrid production plant
- Spent fuel pool, challenges grace time, time to uncover, repair and triggers

4.1. Availability Assessment of Offshore Wind Farm

Assessment of the reliability or availability of an electrical distribution can be relevant to study to understand the operational risk. This is exemplified by assessments of the electrical grid of for example an offshore wind farm, see figure 3. The aim of the analysis is primarily to understand the availability and output that the grid can deliver. The challenge is to consider the failures in the system, and the reconfigurations needed to continue operation and then the repair time until the full production can be restored. A specific challenge is the delay in repair that will be caused since a repair ship will need to be sent. Especially, optimized strategies for when the repair ship shall be sent under the constraint of costs can be developed. This example is presented in a paper by EDF [9].

4.2. Availability Assessment of a Hybrid Production Plant

This example shows a power plant for hybrid energy production with a complex control logic prioritizing between different power sources and possibilities to store the excess energy [10]. The types of components in the knowledge base are presented in Figure 4:

- a consumer representing the electricity demand function that varies over time
- a set of renewables, i.e., wind turbines and solar power plants producing power depending on weather
- a set of backup gas turbines
- batteries or other storage capabilities for storing excess electricity
- a power station controlling the production and connecting all power production systems

Models built with this knowledge base are on a very high level, abstracting away from details in components. The complexity is hidden in the control logic of the plant, how it collects necessary information, takes decisions and propagates them through the plant. The availability analysis for this model needs to resort to Monte Carlo simulations. This, on the other hand, gives full flexibility in defining availability criteria. We can, apart from the fraction of time where the plant fulfills the demand, also analyze costs and additional parameters such as secondary production or over-production. All plant properties that can be expressed in the tool modeling language about the plant can be measured by the Monte Carlo simulator.

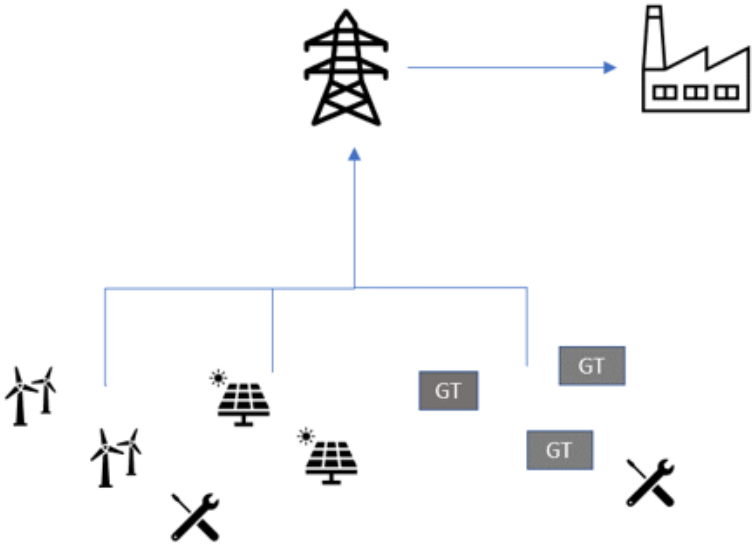

Figure 4. An example of a hybrid production plant

4.3. Assessment of Spent Fuel Pool

Spent Fuel Pool (SFP) analysis differs from a normal nuclear PSA in several aspects. The whole plant, depicted in a simplified form in Figure 5, is much simpler with fewer redundant backups. The SFP Cooling System is constantly running and its potential failures can be tolerated for a relatively long period of time [16]. This gives additional possibilities for repairs of the failed components.

This plant is suitable for a high-level knowledge base which includes components for thermohydraulic systems. A single knowledge base can include behaviors required in different types of analyses [15]. We can generate a fault tree model for systems in this plant, such as the SFP Cooling System, and perform an availability assessment by estimating the unavailability, the unconditional failure intensity and the mean time to repair of the system. This simplifies the standard, fault tree-based, availability analysis by the fact that we can mode the system on a high level and generate all fault trees automatically.

At the same time, we can switch to a dynamic interpretation of the model and generate a low level description based on Markov Processes. An analysis of this spent fuel pool model based on Monte Carlo simulations can estimate various availability criteria or the frequency of uncovering the fuel for scenarios defined by specific plant configurations, reliability data, repair strategies, etc. This type of analysis considers the grace delay between the cooling failure and the consequence in a way that is much more realistic than a static analysis with a fixed mission time. The price to pay is the analysis time required for simulations and the lack of exhaustiveness, which is offered by dynamic methods based on minimal cut sets.

5. FLEXIBILITY OF KNOWLEDGE ENCAPSULATION

The examples in the previous section demonstrate the wide applicability of Model Based Safety/Availability Assessment, specifically the RiskSpectrum ModelBuilder tool. This breath of the component or plant features that can be modeled in a knowledge base shows that the underlying formalism for specifying component behaviors – the modeling language Figaro [17,18] – possesses a great expressive power. Mostly, there are no atomic, predefined components that one must use to build custom components. Properties, linking to other components, behavior including failures and propagating of failures through the system are specified from scratch in the text-based modeling language.

The other property of Figaro which provides a unique expressive power is the first order quantification over related objects. It is possible to express conditions like ‘if all upstream components of this component that have been already started are functional then…’ or ‘if there is a component in the electric support system of this component that has failed then…’. Propagating event effects can utilize the same expressive power to select objects that shall be updated.

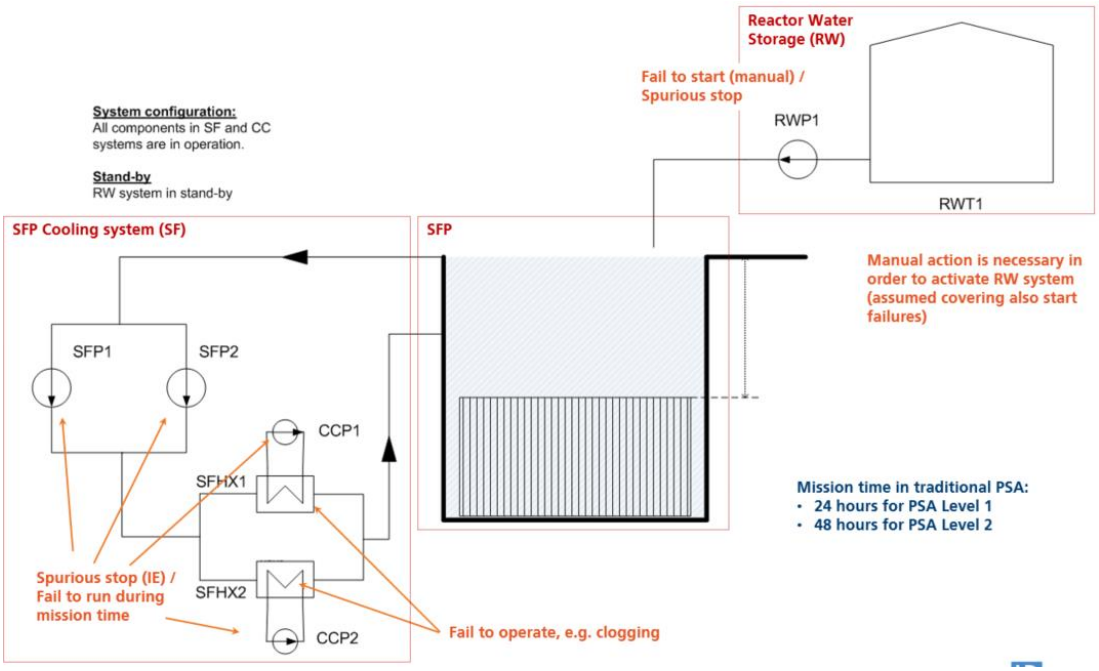

Figure 5. A simplified schematic description of a Spent Fuel Pool

5.1. Strategies for verification of encapsulated knowledge

The expressive power of Figaro comes at a cost. A compact description of a component might hide mistakes. It is therefore essential to validate a knowledge base before it is used in reliability or availability studies. Here we list a couple of best practices for development and verification of a knowledge base.

1. No code duplication: this is, strictly speaking, not a validation rule, but it greatly simplifies it. Whenever one writes code for the same behavior again at a different place, it is time to restructure the classes and abstract this behavior into a more general class. This will typically not require any updates in the already developed studies.
2. Naming conventions: clear and unique naming of class elements will avoid unintended name clashes and simplify inspection of object states during the validation process.
3. Documentation: comments in classes or documentation in a separate document helps to discover design issues in an early phase and makes maintenance of the knowledge base easier in the long run. It also helps in knowledge transfer to new colleagues.
4. Visualizations: a part of the knowledge base definition is a graphical representation of objects together with displaying selected properties. For each class that corresponds to a component visible in RiskSpectrum ModelBuilder, we should define at least one visualization mode that will show all properties relevant for a quick visual evaluation of the component state.
5. Testing of classes in isolation: this resembles unit testing in programming. For each behavior of a class, we should create a test that will activate it and verify that the outcome is correct. This test will have a form of a small model that includes the tested component and triggers the tested behavior. In the ideal case, the test does not include anything not relevant for the studied behavior and will be very small, say less than 10 components. There are two possibilities of inspecting the outcome:
    a. Interactive simulations: the tool offers a possibility to trace the behavior of the modeled system step by step, always selecting the next event that occurs. After each step, we can inspect the state of each class either by means of a visualization or by exploring the internal properties of an object in a special window. By this, we can trigger the tested behavior and verify that the outcome is correct.
    b. Generating fault trees or running a Monte Carlo simulation: this is the ultimate test of behavior correctness. Once we have inspected the component by interactive simulations, we can employ it in an analysis adequate to the knowledge base. If the purpose is to generate fault trees, we can generate fault trees and either manually or by fault tree analysis verify that the logic is as expected. If the purpose is to run Monte Carlo simulations, then we can

define properties to be measured which are relevant for the correctness argument and run simulations.

6. Adding test mode to class definitions: having a possibility to tell an object that it is now a part of a testing setup and that it shall behave in a certain way simplifies building test cases. Sometimes, the default behavior of a component expects that it is a part of a larger network and all links to other components are defined. Only then it can exhibit certain behaviors. This might be impractical in setting up test cases, because we would need a large number of objects. Having a switch that triggers certain behavior even without all linked objects defined helps building small test cases.
7. Testing the new or updated knowledge base on a small, well-defined system. Testing the behavior on a well-defined system where the outcome is known or expected will be the final test before going into production.

A very useful feature is that knowledge bases are reusable. You do not have to start from scratch. Once confidence has been built up for a knowledge base, the complete process will not have to be performed again. Only improvements will have to be verified, and relevant part of regression tests. Very much like software development of new features in an existing framework. The quality assurance will be gained over time and can be reused. It could also be possible to use an already existing knowledge base someone else has developed as a starting point, to reduce the quality assurance of the knowledge base (of course assuming that you have confidence in the organization that developed the knowledge base).

6. CONCLUSION

This paper discusses the use of Model Based Safety Assessment and how that can not only participate but have a leading role in a digital transformation that should take place in the generation of PSA models. We briefly consider reasons for the currently low use of MBSA approaches where two main reasons are anticipated to be habits and deadline driven development.

Some situations where MBSA approaches are superior to the existing modelling approaches are outlined, and a few examples of availability analyses are provided. Availability assessments can surely be performed by tools dedicated to a specific purpose, so the paper also outlines why MBSA approaches are preferable – the possibility to tailor make the basis, the knowledge base in RiskSpectrum ModelBuilder, such that the properties needed are exactly the ones needed to perform the specific type of assessment to be conducted.

The flexibility of an MBSA approach using an encapsulated knowledge, a knowledge base, comes with a price. The knowledge base needs to be quality assured. In the paper we propose a strategy to quality assure the knowledge base, a strategy that shares some practices with software development.

References

[1] Coudert O., Madre, J.-C. Fault Tree Analysis: 1020 Prime Implicants and Beyond. In Annual Reliability and Maintainability Symposium, 1993.
[2] Rauzy A., New Algorithms for Fault Trees Analysis. Reliability Engineering & System Safety 59(2): 203–211. 1993.
[3] Krcal P., Wang P., Bäckström O. Implementation of Conditional Quantification in RiskSpectrum. In Proc. of PSAM 16, 2022.
[4] Nathan E. Wiltbank, Camille J. Palmer. “Dynamic PRA Prospects for the Nuclear Industry”, Frontiers in Energy Research, 2021.
[5] IBM web page on July 2024: www.ibm.com/topics/digital-transformation,
[6] Dirksen G., Kancev D., Shevchenko A., Bell C., Kollasko, H. Creating a Digital Twin Reliability Model Using RiskSpectrum ModelBuilder. In Proc. of ESREL’22. 2022.
[7] Kancev D., Dirksen G., Hausherr R., Shevchenko A., Bell C., Kollasko, H. Development of a new plant-specific, full-scope industrial-scale L1/L2 PSA-model with the application of the new RiskSpectrum® Model Builder Tool. In Proc. of PSAM16, 2022.
[8] Bos W., Volk M., Krcal P., Bouissou M., Stoelinga M. Inferring Piping and Instrumentation Diagrams from Fault Tree Models. In Proc. of ESREL’23. 2023.

[9] Bouissou M.: A Benchmark on Reliability of Complex Discrete Systems: Emergency Power Supply of a Nuclear Power Plant. In Proc. of MARS, EPTCS, pp. 200–216. 2017.
[10] Bouissou, M., Khan, S., Katoen J.-P., Krcal P. Various Ways to Quantify BDMPs. In Proc. of MARS'20, 2020.
[11] Bouissou M., Bon J. L. A new formalism that combines advantages of fault-trees and Markov models: Boolean logic driven Markov processes. Reliability Engineering & System Safety 82, 149-163, 2003.
[12] Bouissou M. Personal communication. To be published at the Lambda Mu 2024 congress.
[13] Chaudonneret T. et. al. K6: a tool for studying reliability of industrial power networks at the 20th Lambda-Mu Conference, Saint-Malo, France. 2016. (paper is in French, but name is a translation)
[14] Krcal P., Troili H., Bäckström O. Control Logic Encoding using RiskSpectrum ModelBuilder, In Proc. of PSAM16, 2022.
[15] Bäckström O., Bouissou M., Krcal P., Wang P. Flexibility of Analysis Through Knowledge Bases, Proc. of ESREL'21, 2021.
[16] Olofsson F., Olsson A. Challenges and lessons learned from a PSA on a spent fuel pool facility. In Proc. of PSAM16, 2022.
[17] Bouissou M., Bouhadana H., Bannelier M., Villatte N. Knowledge modeling and reliability processing: presentation of the Figaro language and associated tools, Proc. of SAFECOMP'91, 1991.
[18] Bouissou M., Humbert S., Houdebine J. Reference manual for the FIGARO probabilistic modelling language (Version-E), EDF, 2019