# BEEHIVE - Behaviour-Induced Configuration of High Variability-Intensive Systems

Valeria Trombetta
valeria.trombetta@uni.lu
SnT - University of Luxembourg
Luxembourg, Luxembourg, Luxembourg

## ABSTRACT

Nowadays, Cyber-Physical Systems (CPS) represent one of the main core elements of the Industry 4.0. It is common practice to run simulations on a model of the CPS, by adopting specific tools and approaches. Since the purpose of such models is to represent real systems, it is appropriate to assume that several components may be affected by noises and disturbances (N&D), and that these latter may have a different impact on the system depending on the considered configuration and the simulation scenarios. The analysis of signals belonging to a CPS system permits the understanding of the relationships that discipline the behavior of the whole system in presence of N&D. Depending on the context and the considered scenarios, the simulations in presence of N&D might generate very different numerical results compared to the simulations that do not include them. However, the simulations with additional N&D are non-trivial to be computed and analyzed, especially when the considered CPS have also high variability and configurability. The adopted approach investigates the validation of possible cross-configurations, in order that the solution includes sets of suitable configurations for both the CPS parameters and N&D wrt scenarios.

## CCS CONCEPTS

• **Software and its engineering** → *Software verification and validation*; • **Computing methodologies** → **Modeling and simulation**; • **Computer systems organization** → *Embedded and cyber-physical systems.*

## KEYWORDS

Simulation, Cyber-Physical Systems, Verification

**ACM Reference Format:**
Valeria Trombetta. 2022. BEEHIVE - Behaviour-Induced Configuration of High Variability-Intensive Systems. In *26th ACM International Systems and Software Product Line Conference - Volume B (SPLC '22), September 12–16, 2022, Graz, Austria.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3503229.3547064


*SPLC '22, September 12–16, 2022, Graz, Austria*

ACM ISBN 978-1-4503-9206-8/22/09.
https://doi.org/10.1145/3503229.3547064

## 1 INTRODUCTION AND MOTIVATION

Multiple modern software systems have to manage a variety of contexts, including new customer requirements and changing environmental conditions. It is costly and inefficient to develop of a new variant of a system entirely from scratch all the times that there is a previously-unseen context. To obtain a better adaptability, compliance and economies of scale, engineers normally inject variation points in the system, which can then be adapted to the precise context newly met. As a result these systems are named Variability-Intensive Systems (VIS). VIS are referred to a large class of systems that can be derived into multiple variants including software product lines ([37]) and configurable systems ([49][45]). The concept of variability is related to each fashion in which the variants are not equal, including, for example, a different value for a variable. In software product lines, such variation points are typically called features, and in configurable systems they are named configuration parameters. Taking into account desired scenarios, engineers have to build, deploy and run suitable variants of their VIS, in order to ensure the satisfaction of the desired requirements. To obtain this, they begin the development of a configuration process in which the configuration parameters (i.e. the variation points) of the VIS are set to specific values to get the opportune variant ([49][18]. In multiple embedded systems (i.e. real-world cases), the presence of variability affects the system behaviour in various ways, and due to this condition the assessment of variants w.r.t. intended requirements is not trivial. In addition to this the injection of variability may lead to an intractable situation due to the exponential computation as consequence to the presence of many millions of variants ([8]). Furthermore since VIS may be related to multiple domain the identification of suitable variant is even more complex as there are multiple criteria and constraints to be taken into consideration. Moreover the choice of the appropriate variant is even more challenging in case of uncertainties (e.g. noises, disturbances, etc.), as the requirements of the system should be satisfied with the highest probability in the majority of contexts ([48][41]) but the behavior of the system is more unpredictable in these conditions. In VIS the variability can be related to the design-time and the run-time. In the first one the requirements have been a-priori defined by engineers, thus the aim is to discover which variants of their VIS are more likely to satisfy the requirements in the best fashion and then which ones should be built/deployed. In the second one, the system (i.e. a specific variant of the VIS) is already running but is regularly faced to unpredictable changes in its environment. To monitor that the requirements are still satisfied in spite of the changing environmental conditions, engineers have to reconfigure the system, i.e. switching the latter from a variant to another one by altering the value of its configuration parameters during its execution. This is why the development of an approach that can handle these two kind of variability is not trivial. The preliminary work presented in this paper is focused on the analysis of Cyber-Physical Systems (CPS) [29, 43] and their behavior in presence of variability related to parameter configurations and uncertainties (N&D). The goal of the research is to help engineers to explore all the possible and appropriate design alternatives wrt the scenario, the mission and the budget. CPS are related to various domains including aerospace [67], automotive [40], avionic [50], healthcare [28], transportation [21], industrial production [36, 63], environmental [35], etc. In this work, we take into account existing Nasa aerospace systems [1], which are extremely critical and are intensively subject to environmental perturbations. Due to their critical missions it is fundamental that CPS act correctly and do not behave unexpectedly. This is the reason why the design, construction and verification of these systems are important steps throughout which quality should be ensured. These steps, however, are challenging due to the difficulty to represent and emulate the various and complex system-world interactions [12, 24, 56]. The quality is an also an essential element to be proved since it is related to the satisfaction of several safety industrial standards. Given a mission and a scenario the satisfaction of safety standard (i.e. the quality) is the core element to the development of a system that could exists in reality. CPS engineers typically rely on simulations and related platforms, such as MathWorks' Simulink, to prepare and assess candidates designs for their system. Simulink is a de-facto industry standard [57] well known by engineers, and it is an environment based on block diagrams which permits the modeling and simulations of dynamical systems. The tool developed to perform preliminary analysis and experiments is easy-reusable and extendable. It is also entirely developed adopting the notorious MathWorks software products (i.e. MATLAB and Simulink), so there is no need for engineers to gain expertise with additional frameworks.

## 2 RESEARCH QUESTIONS

Given a set of simulation scenarios, the purpose of the research is to identify the best configuration for the mission of each specific considered scenario, the desired assurance level, as well as the budget which should be minimized. In order to pursue such an aim, the following research questions have been investigated: **RQ1) Can we discover alternatives of real-world designs that behave correctly? RQ2): Do these design alternatives remain valid in the presence of noises and disturbances? RQ3): Do the design alternatives have the same trends of behaviour as the original system?** Additional research questions that are currently under investigations are: **RQ4) Given a set of scenarios, can we identify the optimal configuration wrt the budget and the desired assurance level? RQ5) Is it feasible to develop an approach to reduce the computational time related to the simulations? RQ6) Is the approach suitable also for systems with run-time variability?**

[1]The two considered Nasa models are available at: www.tiny.cc/559tuz and www.tiny.cc/659tuz

## 3 RESEARCH METHODOLOGY AND APPROACH

The main goal is to discover variations of existing designs that exhibit a suitable behaviour. To achieve this, we have implemented an approach which exploits the variability to discover possible design alternatives and configurations. The idea is to identify the best configuration for a specific scenario and the budget. Then the aim is to verify if the selected configurations are still valid also in the presence of uncertainties such as random N&D, in other words if still it satisfies the selected assurance level and the budget or if other kind of configuration should be adopted for such a purpose. To achieve this, we have followed and implemented the analysis procedure which consists in the following steps: 1) selection of the Simulink model and parameters, 2) simulation of the original models, 3) selection of new parameters range, 4) simulations with new parameters ranges, 5) steps 3) and 4) are repeated if simulation errors are not present, otherwise the flow continues (i.e. limits on range parameters are found), 6) selection of new parameters ranges with N&D, 7) simulations, 8) steps 5) and 6) are repeated if simulation errors are not present, otherwise the flow continues (i.e. limits on range parameters w.r.t. N&D are found), 9) simulation campaigns, 10) results and evaluation [2]. In details, when the Simulink models are imported, a nominal simulation (i.e. a standard one with original settings and configuration without N&D) is performed in order to have a reference for next simulations (i.e. the ones with N&D as well as different configurations). Then suitable ranges of model's parameter are identified altering the original values by ±5%. This process is repeated until some simulation errors are triggered as this would indicate that the maximum possible ranges (wrt to various physical constraints of the model and the scenario) have been identified. We consider correct all the simulations that end without errors and our approach ensures that only valid ranges are considered since the ones that lead to errors are discharged. This process is repeated when N&D are injected in the model. Noises are modeled as white ones (i.e. noises with a zero mean). When noises are enabled, they act on sensors, and their effect is present during the entire simulation. We implement their injection through a random seed. We limit the noise injected into each element by a specific percentage that is model dependent. Disturbances are modeled as random values (w.r.t. tailored ranges) added to the signal of the variables. During the simulation, disturbances can occur in multiple random moments and time ranges. The precise time ranges and the amplitude of the disturbances are randomly determined. In order to control these effects, we specify a maximum range of disturbance that is model dependent. This is why an additional tailoring phase is required, in fact the parameters ranges identified in the previous phase are reduced due to the presence of N&D.

## 4 PRELIMINARY RESULTS

For both the models the computed results are interesting as, all the configurations overall keep the general trend of the original Simulink model. The differences that are present between the nominal versions and the ones with configurations are due to random values related to the variables, noises and disturbances (these latter

[2]The visual representation of the approach as well as supplementary material, such as examples of disturbances, are available at: www.tiny.cc/ai5tu

appear in a random amount of times during simulations and have even random amplitudes). All of these random values are computed w.r.t. the tailored ranges, but in multiple configurations there can be many variables which can assume values close to the limit of the range. This same condition can be present also for N&D and all of these can affect the final results. To answer to **RQ1)** our investigations have revealed that it is possible to discover alternatives of real-world designs that still behave correctly. Expecially for one of the two Nasa models under investigations the range of configurations for some variables are even the double of the other model. This indicates that one of the two models is more sensitive to the presence of configurations and N&D. Having identified the set of valid configurations, we next proceed with the analysis for these configurations with the injection of N&D. This process requires the variables to be afflicted by N&D and the setting of a maximum noise and a maximum disturbance to be injected in the studied models adopting two different extents. This difference exists because we empirically determined that the one of the two Simulink models is more sensitive to N&D than the other one. This also corroborates our previous finding that this latter's parameters have a larger range of configurations than the other model, i.e. the first has a larger range of possible valid configurations. This answers the **RQ2)** as the second tailoring phase ensures that the remaining ranges are appropriate also taking into account the presence of N&D. Concerning the **RQ3)** the Wilcoxon signed ranked paired [47] and the Vargha Delanay [62] tests are computed. Given a couple of configurations, the first test is adopted to check if the difference between variables's values is statistically relevant. In such test, if the outcome is zero, Vargha Delanay test is not elaborated as it would imply that we fail to reject the null hypothesis (i.e. there is not statistical relevance), otherwise Vargha Delanay test is performed. For such tests there are evaluated the values of the nominal simulation with the ones having configurations with N&D. The goal of Vargha Delanay test is to measure the scientific significance. Through our statistical investigation we discovered that a part of the configurations is different and that this depends mainly on the configuration of parameters than on the presence of N&D. This is mainly related to the one of the model, the more tolerant one wrt configurations and N&D, but considering that such model have ranges that for some variables are even the double of the ones elaborated for the other model, the presence of a part of configuration that is not equivalent to the original one is not an unexpected result. In addition to this in general depending on the key performance indicator (KPI) that are considered relevant for a specific mission, this do not necessarily imply a negative result, as this was verified only for some variables. The procedure is useful to discover which variables are more sensitive to the choice of configurations as well as to the presence of N&D. Such variables in fact can be considered relevant only in some specific kind of missions. Some threats to the validity to the current work include the fact that we considered valid simulations depending on the outcome of Simulink simulations, namely on the presence of errors (e.g. if physical constraints are violated etc.) so we cannot state if these design alternatives could be applied in reality (e.g. due to constraints related to ISO standards, etc.) for this specific scenario. The purpose of this preliminary work was to show the existence of a wide variability space, and it will be part of future work to validate the experiments considering various safety standards, as well as additional missions, scenarios and run-time variability.

# 5 WORK PLAN

We plan to answer research questions **RQ1)**, **RQ2)**, **RQ3)** and **RQ6)** by performing experiments using various specific safety industrial standards, additional missions and scenarios, by testing the validity of BEEHIVE for handling variability at design-time and run-time. The final goal in fact consists in developing an approach that can cope with these two types of variability. Future extensions of this work should address the adoption of further models, in particular, tailoring different ranges concerning configurations, additional, as well as the application of additional domains other than aerospace. Engineers use white noise as a noise that alters an input so that an output is a function of the input and the noise [32], but also additional kind of noises and disturbances will be part of future developments as well as multiple way to inject them in Simulink models. Future studies will consider experimenting on other methodologies to model configurability in CPS and to compare them to the approach adopted in this work. In order to answer to **RQ4)** the preliminary experiments have shown that given a budget and a level of assurance it is possible to find the optimal configuration wrt all of these. Concerning **RQ5)**, currently it is under development an approach based on simulation snapshots. The plan consists in establishing similarity threshold that checks the values of variables belonging to different configurations, if these values are equal or less than the established threshold, the simulation is stopped. Since simulations are extremely time consuming, the idea is to avoid to run entirely all the simulations related to configurations that are too similar and at the same time to explore all the configuration space. For the following months this will be the adopted working plan: *Sep.-Oct. :* enhancement and automatization of the snapshot approach *Nov.-Dec. :* experiments and comparison with other approaches. *Jan.-Feb. :* experiments on additional models (e.g. if possible there will be adopted industrial models belonging to other domains or multi-domains, etc.). *Mar-Apr. :* extension of the approach to cope with run-time systems. *May-Jun. :* experiments on run-time systems and comparison with other approaches for such kind of systems.

# 6 RELATED WORK

## 6.1 Variability modeling with Simulink

All of the following works make use of Simulink for variability modeling, but they mainly address the system configurability and do not consider any N&D. Alalfi et al. [1] have empirically derived five variability operators for Simulink models. Leitner et al. [34] have enhanced the variability by adopting layers of abstractions and an extra binding time for Simulink models. According to the survey elaborated by Berger et al. [7] as many as 38% of respondents have used a home-grown domain-specific tool, including Simulink, to perform activities related to the variability modeling in industrial practice. Basit and Dajsuren [6] proposed a clone management framework to handle variability in Simulink models, by considering both the variability and the functionality. Schlie et al. [52] proposed an holistic approach for the reengineering of an entire Simulink model portfolio into a single variability model. Schulze et al. [55] described the problem of intermixing of various function variants with the variability switching logic adopting a Simulink model. Puhlmann et al. [44] have developed an approach to represent variation points, process variants and variability mechanisms in Simulink. Schlie et al. [51] adopted a technical feature model that reflects the realization of artifacts and their variability in the context of Simulink models. Wille et al. [64] used a novel generated delta language to encode the variability between the variants in delta modules. The approach was validated with Simulink models. Schlie et al. [53] proposed an automatic procedure aiming to detect and cluster variation points in Simulink models. Jongeling et al. [30] developed an approach on the co-evolution of Simulink models in a model-based product line. Ali et al. [2] developed a research plan to build a Simulink-based framework for reasoning about hazards in the automotive domain taking into account the variability that arises from different sources. Wippenbeck et al. [65] developed a toolbox which can enable efficient execution and analysis of Simulink power system model parameter and structure variations. Arrieta et al. [4] proposed a methodology for mutating configurable Simulink models where their variability is expressed as feature models. Schlie et al. [54] enhanced their previous variability mining method in Simulink using a user adjustable similarity metric as well as a novel comparison procedure called matching window technique. Kolassa et al. [31] compared and evaluated variability modeling concepts in the automotive domain using Simulink. Several other works [23, 25, 66] applied the SPL method to the aerospace domain, as well as to all the other domains, but none of those focused on the tolerance of N&D in CPS that is addressed in our work. Finally, Haber et al. [27] applied the method of delta modeling [14] to the Simulink environment in order to obtain a modular and flexible variability modeling approach suitable for Simulink models; however, the aim of our work is not to adapt concepts from SPL such as the delta modeling to Simulink, but directly use Simulink for the modeling of CPS variability. Regarding the use of Simulink as a tool, authors such as Bressan et al. [13], who developed a tool for specifying variability in safety-critical systems and which can produce the correct system configuration models, report that Simulink-based variability management approaches do not support safety annotations. Their approach focuses on the reduction of the gap between domain variability abstractions and functional safety annotations attached to system models. However, our work does not require the use of safety annotations. Deatcu et al. [20] analyzed the differences between variant manager interfaces (based on Simulink) and the Python-based extended system-entity-structure/model-based infrastructure on modeling and simulating variability systems. They suggest to use the latter as it is easier to use for beginners; however, our work is targeting experienced scientists or professionals who should not have any issues in using the Simulink environment. Other approaches, such as those reported by Steiner and Masiero [59], use Simulink only to a limited extent. The variability is handled by additional tools such as pure::variants [9][10][11] and Hephaestus [19] [39] [61].

Other related works are those regarding the application of software product line (SPL) methods [46] to CPS. These are extensively applied in software engineering but not in the aerospace domain where similar approaches are scarce [25]. The SPL engineering paradigm promotes systematic reuse and a-priori identification of variation points in order to make the development of software variants more effective. Our work does not try to adapt SPL to CPS models in the aerospace field but uses Simulink, a common tool already used in both SPL and the aerospace domain, to model and analyze the effect of variability using a cross-configurations approach. To the best of our knowledge, there are not other methods oriented to the configuration of both CPS elements and perturbations, because all the surveyed approaches focus only on the first one.

## 6.2 State-of-the art on Behavioural Verification of VIS

Model checking [33] is one of the most famous techniques to analyse system behaviour wrt requirements. Usually, it takes as inputs a state machine (i.e. an executable model of the system) to be verified and a logic formula that embeds the requirements that the system is expected to satisfy, and it outputs false and counterexamples if the executions of the model did not satisfy the logic formula, true otherwise. The model-checking problem for VIS is more complex than for single systems since each variant must be verified wrt the formula [16]. A possible solution consists in adopting a single-system model checking to all the variants in a separate way. These procedures are called product-based in the software-product-line jargon [60], but due to their monitor of each variant individually they may lead to the exponential blow-up induced by variability. To avoid this complexity, family-based [60] model-checking approaches were developed [16][5]. Their goal is based on the decrement of the verification effort by taking into consideration the commonalities that are present in multiple variants. The analysis is executed on each variant at the same time, using approaches such as late splitting and early joining to check only once a (part of) execution common to multiple variants. In this way they reduce the exponential blow-up, despite it can still be present [15][17]. Feature-based model checking[42] [22] relies on the same goal of avoiding unnecessary computations. These approaches consider that variation points are compositional and decrease the analysis of one variant to the individual analysis of its variation points. This consideration is valid only for VIS that are structured in particular fashions and do not generalize. Family-based and feature-based approaches were taken into account to analyse the behaviour of stochastic VIS [48][41], but they are not scalable or they have strict assumptions on how variation points can affect the system, and this is valid only in few particular cases. Sample-based methods are the trade-off between the product-based and feature-based ones [60] [3]. They analyse variants separately but they consider also a degraded form of compositionality across features, such that the results for the sampled variants can facilitate the inference of the characteristics of the non-sampled ones. Such goal is reached since the behaviour of variants that are not known is already considered by the ones that are known or via extrapolation based on prediction models [38][26][58].

## ACKNOWLEDGMENTS

This project is supported by FNR Luxembourg (grant C19/IS/13566661/BEEHIVE/Cordy).

## REFERENCES


[1] Manar H Alalfi, Eric J Rapos, Andrew Stevenson, Matthew Stephan, Thomas R Dean, and James R Cordy. 2019. Variability Identification and Representation for Automotive Simulink Models. In *Automotive Systems and Software Engineering*. Springer, 109–139.

[2] Shaukat Ali, Paolo Arcaini, Ichiro Hasuo, Fuyuki Ishikawa, and Nian-Ze Lee. 2019. Towards a framework for the analysis of multi-product lines in the automotive domain. In *Proceedings of the 13th International Workshop on Variability Modelling of Software-Intensive Systems*. 1–6.

[3] Sven Apel, Alexander Von Rhein, Philipp Wendler, Armin Größlinger, and Dirk Beyer. 2013. Strategies for product-line verification: case studies and experiments. In *2013 35th International Conference on Software Engineering (ICSE)*. IEEE, 482–491.

[4] Aitor Arrieta, Urtzi Markiegi, and Leire Etxeberria. 2017. Towards Mutation Testing of Configurable Simulink Models: a Product Line Engineering Perspective. *Jornadas de Ingeniera del So ware y Bases de Datos (JISBD)* (2017).

[5] Patrizia Asirelli, Maurice H Ter Beek, Stefania Gnesi, and Alessandro Fantechi. 2011. Formal description of variability in product families. In *2011 15th International Software Product Line Conference*. IEEE, 130–139.

[6] Hamid Abdul Basit and Yanja Dajsuren. 2014. Handling clone mutations in Simulink models with VCL. *Electronic Communications of the EASST* 63 (2014).

[7] Thorsten Berger, Ralf Rublack, Divya Nair, Joanne M Atlee, Martin Becker, Krzysztof Czarnecki, and Andrzej Wasowski. 2013. A survey of variability modeling in industrial practice. In *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems*. 1–8.

[8] Thorsten Berger, Steven She, Rafael Lotufo, Andrzej Wąsowski, and Krzysztof Czarnecki. 2010. Variability modeling in the real: a perspective from the operating systems domain. In *Proceedings of the IEEE/ACM international conference on Automated software engineering*. 73–82.

[9] Danilo Beuche. 2008. Modeling and building software product lines with pure:: variants. In *Software Product Line Conference, International*. IEEE Computer Society, 358–358.

[10] Danilo Beuche. 2013. pure:: variants. In *Systems and Software Variability Management*. Springer, 173–182.

[11] Danilo Beuche. 2016. Using pure: variants across the product line lifecycle. In *Proceedings of the 20th International Systems and Software Product Line Conference*. 333–336.

[12] Simon Bliudze, Sébastien Furic, Joseph Sifakis, and Antoine Viel. 2019. Rigorous design of cyber-physical systems. *Software & Systems Modeling* 18, 3 (2019), 1613–1636.

[13] Lucas Bressan, André Luiz de Oliveira, Fernanda Campos, and Rafael Capilla. 2021. A variability modeling and transformation approach for safety-critical systems. In *15th International Working Conference on Variability Modelling of Software-Intensive Systems*. 1–7.

[14] Dave Clarke, Michiel Helvensteijn, and Ina Schaefer. 2015. Abstract delta modelling. *Mathematical Structures in Computer Science* 25, 3 (2015), 482–527.

[15] Andreas Classen, Maxime Cordy, Pierre-Yves Schobbens, Patrick Heymans, Axel Legay, and Jean-François Raskin. 2012. Featured transition systems: Foundations for verifying variability-intensive systems and their application to LTL model checking. *IEEE Transactions on Software Engineering* 39, 8 (2012), 1069–1089.

[16] Andreas Classen, Patrick Heymans, Pierre-Yves Schobbens, Axel Legay, and Jean-François Raskin. 2010. Model checking lots of systems: efficient verification of temporal properties in software product lines. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 1*. 335–344.

[17] Maxime Cordy, Andreas Classen, Patrick Heymans, Axel Legay, and Pierre-Yves Schobbens. 2013. Model checking adaptive software with featured transition systems. In *Assurances for Self-Adaptive Systems*. Springer, 1–29.

[18] Maxime Cordy and Patrick Heymans. 2018. Engineering configurators for the retail industry: experience report and challenges ahead. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*. 2050–2057.

[19] André L de Oliveira, Rosana TV Braga, Paulo C Masiero, Yiannis Papadopoulos, Ibrahim Habli, and Tim Kelly. 2014. A model-based approach to support the automatic safety analysis of multiple product line products. In *2014 Brazilian Symposium on Computing Systems Engineering*. IEEE, 7–12.

[20] Christina Deatcu, Thorsten Pawletta, and Hendrik Folkerts. 2019. MATLAB/Simulink's Variant Manager vs SESToPy. *Simul. Notes Eur.* 29, 1 (2019), 39–43.

[21] Lipika Deka, Sakib M Khan, Mashrur Chowdhury, and Nick Ayres. 2018. Transportation cyber-physical system and its importance for future mobility. In *Transportation Cyber-Physical Systems*. Elsevier, 1–20.

[22] Kathi Fisler and Shriram Krishnamurthi. 2005. Decomposing verification around end-user features. In *Working Conference on Verified Software: Theories, Tools, and Experiments*. Springer, 74–81.

[23] Jesús Padilla Gaeta and Krzysztof Czarnecki. 2015. Modeling aerospace systems product lines in SysML. In *Proceedings of the 19th international conference on software product line*. 293–302.

[24] Andreas Gerstlauer, Christian Haubelt, Andy D. Pimentel, Todor P. Stefanov, Daniel D. Gajski, and Jürgen Teich. 2009. Electronic System-Level Synthesis Methodologies. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 28, 10 (2009), 1517–1530.

[25] Chengliang Guo, Junyuan Teng, Meng Gao, and Yunsong Jiang. 2021. The Research of Aerospace Software Product Line Engineering. In *Signal and Information Processing, Networking and Computers*. Springer, 357–364.

[26] Jianmei Guo, Dingyu Yang, Norbert Siegmund, Sven Apel, Atrisha Sarkar, Pavel Valov, Krzysztof Czarnecki, Andrzej Wasowski, and Huiqun Yu. 2018. Data-efficient performance learning for configurable systems. *Empirical Software Engineering* 23, 3 (2018), 1826–1867.

[27] Arne Haber, Carsten Kolassa, Peter Manhart, Pedram Mir Seyed Nazari, Bernhard Rumpe, and Ina Schaefer. 2013. First-class variability modeling in matlab/simulink. In *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems*. 1–8.

[28] Shah Ahsanul Haque, Syed Mahfuzul Aziz, and Mustafizur Rahman. 2014. Review of cyber-physical system in healthcare. *international journal of distributed sensor networks* 10, 4 (2014), 217415.

[29] Liang Hu, Nannan Xie, Zhejun Kuang, and Kuo Zhao. 2012. Review of cyber-physical system architecture. In *2012 IEEE 15th international symposium on object/component/service-oriented real-time distributed computing workshops*. IEEE, 25–30.

[30] Robbert Jongeling, Antonio Cicchetti, Federico Ciccozzi, and Jan Carlson. 2020. Co-evolution of simulink models in a model-based product line. In *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*. 263–273.

[31] Carsten Kolassa, Holger Rendel, and Bernhard Rumpe. 2015. Evaluation of variability concepts for simulink in the automotive domain. In *2015 48th Hawaii International Conference on System Sciences*. IEEE, 5373–5382.

[32] Hui-Hsiung Kuo. 2018. *White noise distribution theory*. CRC press.

[33] François Laroussinie. 2010. Christel Baier and Joost-Pieter Katoen Principles of Model Checking. MIT Press (May 2008). ISBN: 978-0-262-02649-9.£ 44.95. 975 pp. Hardcover. *Comput. J.* 53, 5 (2010), 615–616.

[34] Andrea Leitner, Wolfgang Ebner, and Christian Kreiner. 2013. Mechanisms to handle structural variability in matlab/simulink models. In *International Conference on Software Reuse*. Springer, 17–31.

[35] George Mois, Teodora Sanislav, and Silviu C. Folea. 2016. A Cyber-Physical System for Environmental Monitoring. *IEEE Transactions on Instrumentation and Measurement* 65, 6 (2016), 1463–1471.

[36] Pieter J Mosterman and Justyna Zander. 2016. Industry 4.0 as a cyber-physical system study. *Software & Systems Modeling* 15, 1 (2016), 17–29.

[37] L.M. Northrop. 2002. SEI's software product line tenets. *IEEE Software* 19, 4 (2002), 32–40. https://doi.org/10.1109/MS.2002.1020285

[38] Jeho Oh, Don Batory, Margaret Myers, and Norbert Siegmund. 2017. Finding near-optimal configurations in product lines by random sampling. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*. 61–71.

[39] André Luiz De Oliveira, Rosana TV Braga, Paulo Cesar Masiero, Yiannis Papadopoulos, Ibrahim Habli, and Tim Kelly. 2016. Model-based safety analysis of software product lines. *International Journal of Embedded Systems* 8, 5-6 (2016), 412–426.

[40] Aditya Patil, Kedar More, and Siddhesh Kulkarni. 2018. A Review on Vehicular Cyber Physical Systems. *International Research Journal of Engineering and Technology (IRJET)* 5, 2 (2018), 1138–1143.

[41] Leonardo Pessoa, Paula Fernandes, Thiago Castro, Vander Alves, Genaína N Rodrigues, and Hervaldo Carvalho. 2017. Building reliable and maintainable dynamic software product lines: An investigation in the body sensor network domain. *Information and Software Technology* 86 (2017), 54–70.

[42] Malte Plath and Mark Ryan. 2001. Feature integration using a feature construct. *Science of Computer Programming* 41, 1 (2001), 53–84.

[43] Andr Platzer. 2018. *Logical Foundations of Cyber-Physical Systems* (1st ed.). Springer Publishing Company, Incorporated.

[44] Frank Puhlmann, Arnd Schnieders, Jens Weiland, and Mathias Weske. 2005. Variability mechanisms for process models. *PESOA-Report TR* 17 (2005), 10–61.

[45] Mikko Raatikainen, Timo Soininen, Tomi Männistö, and Antti Mattila. 2003. A case study of two configurable software product families. In *International Workshop on Software Product-Family Engineering*. Springer, 403–421.

[46] Rick Rabiser and Alois Zoitl. 2021. Towards Mastering Variability in Software-Intensive Cyber-Physical Production Systems. *Procedia Computer Science* 180 (2021), 50–59.

[47] Denise Rey and Markus Neuhäuser. 2011. *Wilcoxon-Signed-Rank Test*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1658–1659.

[48] Genaína N Rodrigues, Vander Alves, Vinicius Nunes, André Lanna, Maxime Cordy, Pierre-Yves Schobbens, Amir Molzam Sharifloo, and Axel Legay. 2015. Modeling and verification for probabilistic properties in software product lines. In *2015 IEEE 16th International Symposium on High Assurance Systems Engineering*. IEEE, 173–180.

[49] Daniel Sabin and Rainer Weigel. 1998. Product configuration frameworks-a survey. *IEEE Intelligent Systems and their applications* 13, 4 (1998), 42–49.

[50] Krishna Sampigethaya and Radha Poovendran. 2013. Aviation cyber–physical systems: Foundations for future aircraft and air transport. *Proc. IEEE* 101, 8 (2013), 1834–1855.

[51] Alexander Schlie, Alexander Knüppel, Christoph Seidl, and Ina Schaefer. 2020. Incremental feature model synthesis for clone-and-own software systems in MATLAB/Simulink. In *Proceedings of the 24th ACM Conference on Systems and Software Product Line: Volume A-Volume A*. 1–12.

[52] Alexander Schlie, Christoph Seidl, and Ina Schaefer. 2019. Reengineering Variants of MATLAB/Simulink Software Systems. In *Security and Quality in Cyber-Physical Systems Engineering*. Springer, 267–301.

[53] Alexander Schlie, David Wille, Loek Cleophas, and Ina Schaefer. 2017. Clustering variation points in matlab/simulink models using reverse signal propagation analysis. In *International Conference on Software Reuse*. Springer, 77–94.

[54] Alexander Schlie, David Wille, Sandro Schulze, Loek Cleophas, and Ina Schaefer. 2017. Detecting variability in MATLAB/Simulink models: An industry-inspired technique and its evaluation. In *Proceedings of the 21st International Systems and Software Product Line Conference-Volume A*. 215–224.

[55] Michael Schulze, Jens Weiland, and Danilo Beuche. 2012. Automotive model-driven development and the challenge of variability. In *Proceedings of the 16th International Software Product Line Conference-Volume 1*. 207–214.

[56] Sanjit A Seshia, Shiyan Hu, Wenchao Li, and Qi Zhu. 2016. Design automation of cyber-physical systems: Challenges, advances, and opportunities. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 36, 9 (2016), 1421–1434.

[57] Sohil Lal Shrestha, Shafiul Azam Chowdhury, and Christoph Csallner. 2020. DeepFuzzSL: Generating models with deep learning to find bugs in the Simulink toolchain. In *2nd Workshop on Testing for Deep Learning and Deep Learning for Testing (DeepTest)*.

[58] Norbert Siegmund, Alexander Grebhahn, Sven Apel, and Christian Kästner. 2015. Performance-influence models for highly configurable systems. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*. 284–294.

[59] Eduardo Steiner, Paulo Masiero, and Rodrigo Bonifacio. 2013. Managing SPL Variabilities in UAV Simulink Models with Pure: variants and Hephaestus. *CLEI Electronic Journal* 16 (04 2013), 7 – 7.

[60] Thomas Thüm, Sven Apel, Christian Kästner, Ina Schaefer, and Gunter Saake. 2014. A classification and survey of analysis strategies for software product lines. *ACM Computing Surveys (CSUR)* 47, 1 (2014), 1–45.

[61] Lucinéia Turnes, Rodrigo Bonif, Vander Alves, Ralf Lammel, et al. 2011. Techniques for developing a product line of product line tools: a comparative study. In *2011 Fifth Brazilian Symposium on Software Components, Architectures and Reuse*. IEEE, 11–20.

[62] Andras Vargha and Harold D Delaney. 2000. A critique and improvement of the CL common language effect size statistics of McGraw and Wong. *Journal of Educational and Behavioral Statistics* 25, 2 (2000), 101–132.

[63] Lihui Wang, Martin Törngren, and Mauro Onori. 2015. Current status and advancement of cyber-physical systems in manufacturing. *Journal of Manufacturing Systems* 37 (2015), 517–527.

[64] David Wille, Tobias Runge, Christoph Seidl, and Sandro Schulze. 2017. Extractive software product line engineering using model-based delta module generation. In *Proceedings of the Eleventh International Workshop on Variability Modelling of Software-intensive Systems*. 36–43.

[65] Tilman Wippenbeck, Reinhold Bertram, and Armin Schnettler. 2015. *A toolbox for efficient parameter and structure variation of time domain power system simulation models in Simulink*. IEEE.

[66] Nazish Yousaf, Maham Akram, Amna Bhatti, Ammara Zaib, et al. 2019. Investigation of Tools, Techniques and Languages for Model Driven Software Product Lines (SPL): A Systematic Literature Review. *Journal of Software Engineering and Applications* 12, 07 (2019), 293.

[67] Lichen Zhang. 2013. View oriented approach to specify and model aerospace cyber-physical systems. In *2013 IEEE 11th International Conference on Dependable, Autonomic and Secure Computing*. IEEE, 296–303.