Applying model-based Co-Simulation on modular Production Units in Complex Automation Systems

Christoph Binder
Josef Ressel Center for Dependable
System-of-Systems Engineering
Urstein Sued 1, A–5412 Puch/Salzburg, Austria
christoph.binder@fh-salzburg.ac.at

Adnan Agic
Josef Ressel Center for Dependable
System-of-Systems Engineering
Urstein Sued 1, A–5412 Puch/Salzburg, Austria
aagic.itsb-m2017@fh-salzburg.ac.at

Christian Neureiter
Josef Ressel Center for Dependable
System-of-Systems Engineering
Urstein Sued 1, A–5412 Puch/Salzburg, Austria
christian.neureiter@fh-salzburg.ac.at

Arndt Lüder
Otto-v.-Guericke University
Universitätsplatz 2
D-39106 Magdeburg, Germany
arndt.lueder@ovgu.de

***Abstract*—Contemporary manufacturing systems face major challenges driven by the ongoing integration of intelligent manufacturing components, mainly caused by the emergence of the Industrial Internet of Things (IIoT). As the interplay of those Cyber-physical Systems (CPS) itself forms a System of Systems (SoS), engineering such a system becomes a difficult task. This is mainly attributed to the evolutionary development of its independently operating components, which results in some kind of unpredictable and often undesirable behavior. Considering this from a developer's perspective, it is important to investigate the system's behaviors before its actual implementation. However, as methods applied in Model Based Systems Engineering (MBSE) have proven to be a major technology driver when it comes to developing the architecture of these industrial systems, their simulation is still not standardized. Thus, this paper introduces an Industry 4.0 specific tool-chain enabling the co-simulation of components within any developed industrial system architecture. This allows the investigation of varying industrial CPS and their interplay during run-time with the goal to detect undesired emergent behaviors. By doing so, the model is developed regarding the specifications of the Reference Architecture Industrie 4.0 (RAMI 4.0), while Mosaik is the tool of choice for creating the Co-Simulation environment. In order to validate the developed tool-chain, a case study utilizing modular production units within an industrial automation system is applied.**

***Index Terms*—System Architecture, Industrial Internet of Things (IIoT), Reference Architecture Model Industrie 4.0 (RAMI 4.0), Industrial Systems Engineering, Co-Simulation**

I. INTRODUCTION

Remaining competitive despite continuously enhancing production processes caused by the integration of new technologies is an important goal of most manufacturing companies. The main reason expediting this transformation is the emergence of the fourth industrial evolution with all its dispositions, like the Industrial Internet of Things (IIoT) or Cyber-physical Systems (CPS). For example, new developments in the area of Information and Communication Technology (ICT) allow the integration of more powerful components or an improved communication infrastructure into conventional production systems [1]. On the one hand, these new possibilities in connectivity and interoperability lead to an amalgamation of distinct divisions into a superordinate value creation network, on the other hand more and more industrial agents are added to the production lines, making decisions on their own either in a small frame or on a larger scale [2]. In theory, the classification scheme proposed in [3] can be applied to demonstrate that production lines are complex systems or even a System of Systems (SoS). However, to emphasize the autonomous character of the individual participants contained in these interwoven system structures, the traits *evolutionary development* and *emergent behavior* are utilized as those are difficult to observe in a static modeling environment [4].

With regard to the aforementioned aspects, in order to enable flexibility and adaptability under volatile conditions, several organizations proposed methodologies addressing the increasing complexity in industrial automation systems. Thus, to mention some examples, suitable approaches like the Reference Architecture Model Industrie 4.0 (RAMI 4.0) [5], the Industrial Internet Reference Architecture (IIRA) [6] or the Arrowhead Framework [7] have been developed. Based on these architectural models, extensive methods applying the concepts of Model Based Systems Engineering (MBSE) arose in order to develop IIoT based systems on multiple abstraction levels addressing different stakeholder concerns. More specifically, as a detailed literature research would exceed the scope of this paper, a detailed collection can be found in [8]. As recognizable, the high variety and different aims of the approaches prove that MBSE has become a major technology driver when it comes to design such a system.

However, adversely mostly static aspects of SoS can be considered when modeling the architecture of it. Thus, usually a simulation scenario is applied in order to observe the behavior of multiple industrial agents and their interplay during runtime. Nevertheless, as modeling is already widely used in this area, simulating such a system is still a topic to be formalized.

This results in a lack of suitable tools or companies producing solely proprietary solutions only being able to simulate single aspects or components of the system, like NX introduced by Siemens PLM Software. Thus, in order to close this gap in the Smart Grid area, a co-simulation framework called Mosaik [9] has been introduced, which allows the default investigation of Smart Grid scenarios originating from diverse sources. The flexibility of this framework and the possibility for adaptations could also allow its utilization in the industrial area, as the mentioned aspect would be a requirement for complex automation systems too. However, in contrast to the Smart Grid, where the linear energy flow formulates a standardized problem domain [10], a production system as a large-scale system exhibits multiple manufacturing processes as well as various heterogeneous production systems [11]. The contribution of this paper thus is to validate the general feasibility of the tool-chain for Industry 4.0 scenarios with the goal to investigate undesired emergent behavior in industrial systems, which is done in this paper by especially focusing on modular production units.

By doing so, this contribution is structured as following: in Section II an overview of the related work is given. Subsequently, the scientific approach to validate the feasibility is mentioned in Section III. Section IV deals with the implementation of Mosaik for simulating previously created industrial models. Thus, the feasibility of the tool-chain is demonstrated and thereby validated with a primitive industrial use case in Section V. Finally, in Section VI the results of the conducted study are summarized and a conclusion is given.

II. RELATED WORK

A. Domain-specific Systems Engineering

Systems engineering in the industrial domain is not a completely new topic to talk about. Based on the previously mentioned reference architectures, a number of useful approaches emerged. There is an ongoing discussion on the utilization of reference architectures to improve the production system architecture design [12]. Facing the IIRA and RAMI 4.0, the running CrESt project[1] has a leading position in that field. This project deals with identfying the necessary actions that are required when interchanging models within the mentioned reference architectures. With regard to the IIRA, the authors of [13] proposed an approach for the development of IIoT applications including industrial agents by considering them as SoS. A special feature of their work is the mapping of the IIRA viewpoints to those of the Unified Architecture Framework (UAF). This helps enabling advanced MBSE by applying the features of UAF to the IIRA specific system engineering methods. On the other hand RAMI 4.0 also deals as a template for numerous projects. An example is the work proposed in [14], which introduces modeling the digital twin of a CPS aligned to the specifications of the administration shell originating from RAMI 4.0. Furthermore, another approach is dealing with modeling an industrial agent according to the layers of RAMI 4.0 itself to discover and select equipment for processing operations requested by products [15].

Taking this into further consideration, the authors of [16] extend the existing RAMI 4.0 framework with a methodology for tool-supported systems engineering according to a specific development process. Further specifications resulted in the proposition of an extensive framework called the RAMI Toolbox, inheriting domain-specific standards and containing a lot of functionality. By doing so, this tool utilizes the concepts of the Domain Specific Systems Engineering (DSSE) approach, firstly introduced in [17]. In order to address all aspects of the systems development life-cycle, DSSE is consisted of 8 different steps [17], which include tasks like model checking, visualizing, implementing simulation frameworks or the actual system. However, as most of the mentioned aspects are already established in the industrial area, the simulation of automation systems is still a topic to explore [18].

B. Co-Simulation

Generally spoken, the main goal of simulations is the evaluation of system characteristics like controllability, reliability or its functionality in general. This is done in order to prevent the need to execute resource intensive and dangerous laboratory or field experiments. Unlike other types of simulations, which use either one solver for at least one model or at minimum one solver per model, a co-simulation uses multiple solvers for multiple models [19], [20].

Thus, this type of simulation is optimized to be applied on a SoS, such as a IIoT based system. The main advantage of a co-simulation is the independent operation of each simulator and the possibility to interconnect them dynamically [21], [22]. This interplay however can be enabled by using two different kinds of linking. Either the simulators are individually coupled via interfaces with each other or generically with the help of a certain middle-ware. In the second case, a central unit processes the co-simulation scenario during run-time and deals with exchanging the variables as well as time synchronization, which is an important instrument for securing the mentioned aspects making use of so-called steps to coordinate each simulator as well as the whole simulation scenario [20], [23].

One of those frameworks dealing as middle-ware has been proposed with Mosaik [9]. Originally developed for the Smart Grid area, it is already established and used in several projects [24]–[26]. In contrast to the particularly for the IIoT designed project Avanti[2], which is aimed for virtually commissioning and simulating industrial equipment, Mosaik is in constant development. According to these considerations, Mosaik can be considered as a kind of a hybrid automaton [27].

III. APPROACH

As already mentioned, the goal of this approach is to evaluate whether the well accepted co-simulation concept from the Smart Grid can be transferred and integrated into an Industry 4.0 modeling environment. As the actual result

---

[1] https://crest.in.tum.de/

[2] http://www.avanti-project.de/

cannot be foreseen, a dynamic and changeable method needs to be utilized. Thus, the concepts of the Agile Design Science Research Methodology (ADSRM) seem to be suitable to be applied in this specific scenario [28]. Providing the possibility to enter the development cycle in each of its single steps, the process iteration is usually initialized by choosing a typical use case. Thus, as industrial case study, this example makes use of an actual IIoT application scenario, the transition of the original production line towards modular production units. This example has been selected due to the actual applicability in industrial projects introduced by the German association "Audi" and therefore proves its importance being a representative example for the fourth industrial revolution. Concluding, in Figure 1, the concept of the use case itself is shown, described in [29]. More precisely, a future production plant will be constituted of up to 200 production units, where car bodies are maneuvered through. According to the desired specifications, each car body only visits the production units where configurations need to be made. For example, if the customer did not order heated seats, this unit does not have to be visited, which will approximately result in increased efficiency of around 20%, as mentioned in [29].

As the goal of ADSRM is to start the first iteration in a plain way, a simplified version of the mentioned use case will be applied, with the goal to rather ensure the feasibility than to create a realistic real-world scenario. Thus, a total number of ten specific production units are deployed in this example. In particular, the following units are utilized in this scenario: body shaping, paintwork, chassis, exterior, transmission, engine, electronics, interior, final assembly and quality check. According to this principle, the simulation as well as the developed models are superficial and the results are not optimized, as the first iteration of ADSRM should primarily deal with ensuring the feasibility. Thus, compared to the system's Digital Twin, with this approach, all changes in the model can be simulated immediately and dynamically, allowing to gain fast results for optimizing the system like using another factory layout, new production units or different manufacturing processes. In order to do so, Mosaik itself needs to be validated for its possibility to simulate such an IIoT scenario considering Industry 4.0 requirements like ubiquitous interconnection or manufacturing in lot size 1.

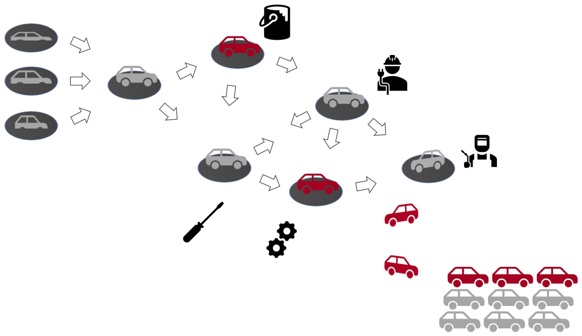

Fig. 1. Modular assembly according to [29]

IV. IMPLEMENTATION

The implementation of the co-simulation scenario itself is split into three different parts, as seen in Figure 2. A detailed description of the depicted process is hence described in the following sections. All developed artifacts are thereby accessible and applicable for reproduction or individual adaptations at the project website[3]. First, the designated case study is modeled according to the architecture of RAMI 4.0, labeled with the letter "A". In the next step, identifiable with "B", the co-simulation environment is set up by adjusting the main scenario file of Mosaik to the intended car manufacturing process of the case study. Additionally, in order to validate the feasability of the Industry 4.0 tool-chain, the functionality of the modeled system components needs to be executed within the co-simulation scenario. This is done by analyzing the structure of Mosaik as well as creating separate Python files representing the simulators, which use the Mosaik Application Programming Interface (API), by executing the following steps. In the process of the simulator generation, illustrated by the letter "C", the modeled scenarios are analyzed towards their functionality and the executable code is implemented within the respective simulators. In the end, the simulators are linked to the core for ensuring a fully functional scenario.

A. Case Study Modeling

The first step towards successfully setting up the co-simulation environment is the development of the case study model. As the use case itself is derived from a complex industrial application, different viewpoints need to be considered. Thus the concepts of RAMI 4.0 are going to support the modeling process and address different aspects of the developed system architecture, like considering the agent's behavior or indicate their interconnection. By following the development process proposed in [16], the requirements ensuring the functionality of each system component are specified in the Business Layer. The Information Layer illustrates the exchanged data while the Communication Layer inherits the corresponding interfaces. Both of the mentioned information is of importance for configuring the co-simulation scenario, as the respective simulators realize the interconnection and data exchange via those interfaces.

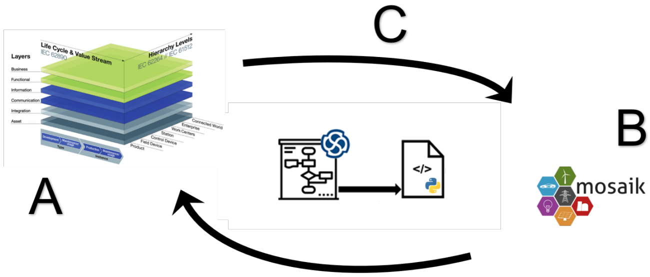

Fig. 2. Scenario Overview

---

[3] http://www.rami-toolbox.org/

Finally, the executable code is delineated within the Function Layer of RAMI 4.0, which is embedded in each simulator and executed by the co-simulation master file. The final result can thereby be browsed through at the project website.

B. System Architecture

The SysML architecture of the simulated system is represented by the modeled case study. It contains several viewpoints giving insights into the production system including requirements, production islands and communication infrastructures. The architecture of the co-simulation environment itself is explained in the following. In order to secure procedural correctness of the simulation during run-time, the so-called *scenario.py* file of Mosaik characterizes the main process. Thereby, the Mosaik framework is initialized by importing the needed modules followed by the creation of entities for each simulator class. The configuration thereby is loaded by a file called *config.json*, where important settings can be placed. The main component of the framework is thereby represented by its core, which deals with managing the respective simulators and ensures a scheduled processing. In this specific use case, the simulators for the ten different production units are instantiated according to the information in the configuration file. This information is previously executed from the SysML model as well as information about the interfaces and data exchange between each of the production units. Those simulators are then added to the simulation scenario during run-time, which allows to simulate various types of control strategies.

C. Simulator Generation

Next, the simulators of the modeled system components are developed. Therefore, each simulator consists of three Python files, an API, the simulator itself and the associated models, explained in detail in the following. While the API deals with implementing the interface of Mosaik, the main purpose of it is to communicate with the co-simulation master algorithm by providing functionalities and data with the help of a Java wrapper. Hence, the main purpose of this wrapper is to provide the Python code to the API of Mosaik. The simulation class itself instantiates all entities of the respective model and administrates its intended functionality. In the models, the business logic as well as the functionality itself is constituted, which the simulator will execute during run-time. However, this is supervised by the configuration of the co-simulation, where the start and end conditions are enabled as well as the treated procedure during each step.

In this case, two different simulator classes are created. The first simulator represents the functionality of the respective production unit. The second class however inherits information about the vehicle to be produced. The functionality of these simulators is thereby quite straight-forward. According to the production state of the car body and the equipment to install, a specific production unit will be approached. On the other hand, the simulator for the production island implements the behavior of the manufacturing process. Moreover, in the step method, the communication between the respective simulators is clearly defined for exchanging their in- and output values. Thereby, in prescribed steps, each car body is treated individually by each production island.

V. APPLICATION

Executing the previously created co-simulation scenario in Mosaik is intended to validate and demonstrate the feasibility of the tool-chain in context of Industry 4.0. Therefore, to limit the complexity of the case study, in this scenario a maximum of 480 time steps and a capacity of 2 available manufacturing spaces for each production island have been chosen. According to the required equipment to be installed for each car, time steps for construction works vary from 1 to 9 time steps. To mention an example, some cars require the additional installation of electronic handbrakes, an adaptive cruise control system and other additional cabling, while some cars only need standard equipment. Randomly generating specifications for each car body will thereby result in different construction times for each single car. In addition, 2 new car bodies are created each 15 time steps, which is initiated by the start of the simulation at time step 0 and ends after 10 cars have been instantiated. Thus, the results of one exemplary simulation run are thereby shown in Figure 3, where the whole production progress is shown by indicating cars under production in orange and inquired ones in blue. Furthermore it is shown that the cars under construction strongly increase at the start of the simulation, which is traced back to the dependencies between the production islands. For example, if a car does not own a chassis, it is not able to install the transmission. At around 80 time steps, this behavior evens out because of the exhausted capacities until it finally decreases caused by the production stop after 10 instantiated cars. Taking the behavior of single production islands into further consideration, a more detailed observation is given. Thus in Figure 4, three different units are shown in different periods of time. More detailed, the first two charts show the comparison between two islands forming the body shape. Due to the randomness of the orders, the transporters either choose one of the shown production islands or the third one. This results in a random allocation to one of the available units. In this case, the early orders required the first body type while the island producing the second type was more occupied halfway of the observation period.

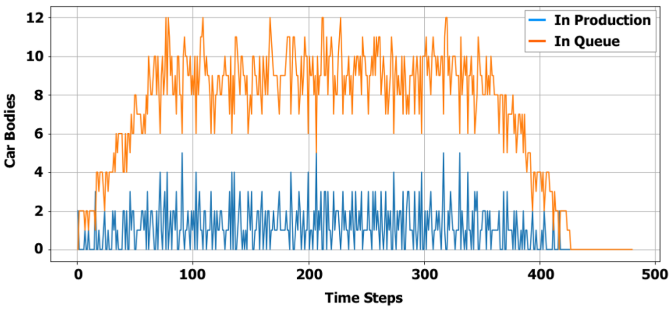

Fig. 3. Exemplary Simulation Run

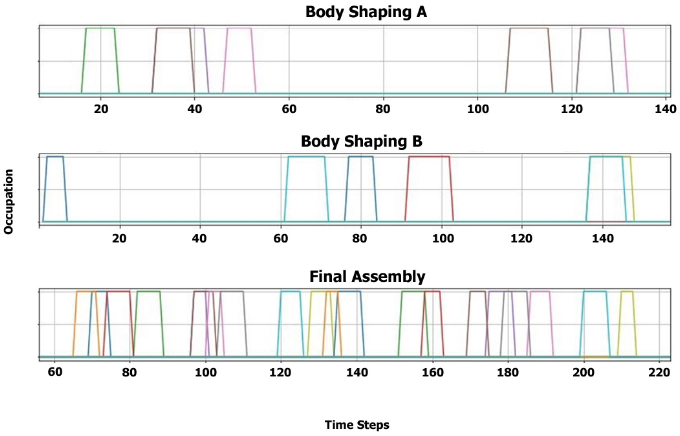

Fig. 4. Exemplary production island occupation

Additionally, the differently colored bars show the time steps the respective production island was occupied while manufacturing the car body, with the different sizes thereby indicating the varying production time. However, the third chart shows that the final assembly unit has been fully utilized all the time except of small breaks.

Additionally, the single time steps a car body has spent at the respective production islands have been recorded. As analyzing each car body by itself would be unnecessarily complex without providing additional value, mean value and standard deviation are assessed. The data is collected from a number of 100 cars that are built with the industrial agents of the modular production line. This means, the data is aggregated from 10 different simulation runs each of them containing 10 cars to manufacture. In order to represent a typical automotive scenario, three different kinds of body shaping islands are introduced. Those are randomly visited by the transporter maneuvering the car bodies, while all other production units are visited by every single transporter. The result is thereby shown in Table I, in which the rows represent the single production islands.

TABLE I
VISIT TIME MEAN VALUE AND STANDARD DEVIATION

Production Island	Amount	Mean Value	Standard Deviation
Body Shaping A	40	8.0	2.16
Body Shaping B	20	7.5	3.35
Body Shaping C	40	6.5	1.29
Paintwork	100	6.3	2.31
Chassis	100	6.7	0.48
Exterior	100	6.4	2.41
Transmission	100	6.4	0.52
Engine	100	7.6	1.71
Electronics	100	4.8	0.92
Interior	100	8.4	1.58
Final Assembly	100	5.8	1.35
Quality Check	100	7.7	0.95

The first column delineates the amount of inspected car bodies, followed by the mean value in the center and the standard deviation in the last column. In detail, the electronics assembly took about 4.8 time steps in average for attaching the electronic parts, while the interior needed 8.4 time steps for successful installation. In addition, the standard deviation reaches from about 0.5 to 2.5 time steps, which is calculated from comparing the mean value of each production island with the result of several independently executed runs.

A. *Findings*

The main goal of this approach, validating the feasibility of an Industry 4.0 tool-chain, has been successfully implemented. The proposed work demonstrates general feasibility that using Mosaik in an Industry 4.0 environment enables the simulation of industrial agents. Thereby, each agent may contain an independent and unique behavior, as realized in the model aligned to RAMI 4.0. This allows to observe their interplay in a large-scale area or evaluate their respective functionality during runtime, which is a big step towards handling the complexity while engineering current or future industrial systems. Compared to other state-of-the-art approaches, a special feature of this work is the flexibility of the contributed approach. In order to investigate the characteristics of a large-scale and multi-agent industrial system, like recognizing emergent behavior, perform a model evaluation or executing unit tests, dynamically configuring single elements could be a beneficial.

Even though the application substantiates the feasibility of the Industry 4.0 tool-chain, the chosen scenario exhibits several limitations. Thus, although being reproducible by applying the uploaded material, the work should not be seen as a ready-to-use methodology. The contrived approach rather validates the applicability of Mosaik in an industrial environment and does not provide any interpretation of simulation results, which could be elaborated in follow-up projects. Thus, even though first indications of emergent behavior can be observed in the simulated scenario, the number of vehicles in the applied case study has been to little to make a meaningful statement. In the future a more sophisticated case study has to be applied to better understand the limitations in a quantitative way.

VI. CONCLUSION & FUTURE WORK

Model-based co-simulation in the industrial area is an almost unexplored area. Most approaches solely address single simulations of system parts or do not consider the behavior of a system in real-time at all. A standardized problem domain and early results from research and development within the Smart Grid area resulted in the emergence of powerful tools like Mosaik. Although being targeted to the power system environment, an application for simulation complex automation systems could be conceivable. Thus, this paper proposes an approach where a system modeled to the specifications of RAMI 4.0 is subsequently simulated with the help of the Mosaik framework. Thereby, first a state of the art analysis shows promising approaches from other organizations and indicates the gaps in this area. In the next step, a suitable IIoT related case study is utilized for this scenario. In order to get an overview of the architecture of the chosen use case, it is previously modeled with the help of the RAMI Toolbox. This helps understanding single aspects and showing the behavior of particular industrial agents. Aiming to validate the functionality of the resulting tool-chain, Mosaik is used for investigating the possibility of simulating an industrial SoS. This is done by implementing the modeled functionality into executable simulators, which are then applied in the co-simulation scenario to investigate industry related features.

Overall, this work should not be seen as a ready-to-use method rather than a initial assessment of suitability. Thus, based on the outcome of this paper, several other research projects could be proceeded. However, at first, the co-simulation framework and interconnection with RAMI 4.0 needs to be improved. Subsequently, to further investigate emergent behavior in such a scenario, a more complex use case needs to be applied. A considerably larger number of production units and more orders in specific time steps would demonstrate such unforeseeable behavior more precisely. Furthermore, to increase the usability of this approach and to implement the concept of Round-trip Engineering (RTE), a bi-directional interface between the RAMI Toolbox and Mosaik has to be established.

ACKNOWLEDGMENT

The financial support by the Austrian Federal Ministry for Digital and Economic Affairs and the National Foundation for Research, Technology and Development and the Christian Doppler Research Association as well as the Federal State of Salzburg is also gratefully acknowledged.

REFERENCES

[1] E. A. Lee, “Cyber physical systems: Design challenges,” in 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC). IEEE, 2008, pp. 363–369.

[2] P. Leitão, S. Karnouskos, L. Ribeiro, J. Lee, T. Strasser, and A. W. Colombo, “Smart agents in industrial cyber–physical systems,” Proceedings of the IEEE, vol. 104, no. 5, 2016, pp. 1086–1101.

[3] R. Haberfellner, O. de Weck,, E. Fricke, and S. Vössner, Systems Engineering - Grundlagen und Anwendung. Orell Füssli, 2015.

[4] M. W. Maier, “Architecting principles for systems-of-systems,” Systems Engineering: The Journal of the International Council on Systems Engineering, vol. 1, no. 4, 1998, pp. 267–284.

[5] M. Hankel and B. Rexroth, “The Reference Architectural Model Industrie 4.0 (RAMI 4.0),” ZVEI, 2015.

[6] S.-W. Lin, B. Miller, J. Durand, R. Joshi, P. Didier, A. Chigani, R. Torenbeek, D. Duggal, R. Martin, G. Bleakley et al., “Industrial Internet Reference Architecture (IIRA),” Industrial Internet Consortium (IIC), Tech. Rep, 2015.

[7] P. Varga, F. Blomstedt, L. L. Ferreira, J. Eliasson, M. Johansson, J. Delsing, and I. M. de Soria, “Making system of systems interoperable–the core components of the arrowhead framework,” Journal of Network and Computer Applications, vol. 81, 2017, pp. 85–95.

[8] A. Wortmann, B. Combemale, and O. Barais, “A systematic mapping study on modeling for industry 4.0,” in 2017 ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems (MODELS). IEEE, 2017, pp. 281–291.

[9] S. Schütte, S. Scherfke, and M. Tröschel, “Mosaik: A framework for modular simulation of active components in smart grids,” in Smart Grid Modeling and Simulation (SGMS), 2011 IEEE First International Workshop on. Brussels, Belgium: IEEE, 2011, pp. 55–60.

[10] C. Greer, D. A. Wollman, D. E. Prochaska, P. A. Boynton, J. A. Mazer, C. T. Nguyen, G. J. FitzPatrick, T. L. Nelson, G. H. Koepke, A. R. Hefner Jr et al., “Nist framework and roadmap for smart grid interoperability standards, release 3.0,” Tech. Rep., 2014.

[11] C. Garrido-Hidalgo, D. Hortelano, L. Roda-Sanchez, T. Olivares, M. C. Ruiz, and V. Lopez, “Iot heterogeneous mesh network deployment for human-in-the-loop challenges towards a social and sustainable industry 4.0,” IEEE Access, vol. 6, 2018, pp. 28 417–28 437.

[12] S. Unverdorben, B. Böhm, and A. Lüder, “Concept for deriving system architectures from reference architectures,” in 2019 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM). IEEE, 2019, pp. 19–23.

[13] A. Morkevicius, L. Bisikirskiene, and G. Bleakley, “Using a systems of systems modeling approach for developing industrial internet of things applications,” in 2017 12th System of Systems Engineering Conference (SoSE). IEEE, 2017, pp. 1–6.

[14] I. Grangel-González, L. Halilaj, G. Coskun, S. Auer, D. Collarana, and M. Hoffmeister, “Towards a semantic administrative shell for industry 4.0 components,” in 2016 IEEE Tenth International Conference on Semantic Computing (ICSC). IEEE, 2016, pp. 230–237.

[15] M. A. Pisching, M. A. Pessoa, F. Junqueira, D. J. dos Santos Filho, and P. E. Miyagi, “An architecture based on rami 4.0 to discover equipment to process operations required by products,” Computers & Industrial Engineering, vol. 125, 2018, pp. 574–591.

[16] C. Binder, C. Neureiter, and A. Lüder, “Towards a domain-specific approach enabling tool-supported model-based systems engineering of complex industrial internet-of-things applications,” Systems, vol. 9, no. 2, 2021.

[17] C. Neureiter, A Domain-Specific, Model Driven Engineering Approach for Systems Engineering in the Smart Grid, Hamburg, Germany, 2017.

[18] M. Schluse, M. Priggemeyer, L. Atorf, and J. Rossmann, “Experimentable digital twins—streamlining simulation-based systems engineering for industry 4.0,” IEEE Transactions on industrial informatics, vol. 14, no. 4, 2018, pp. 1722–1731.

[19] A. M. Law, W. D. Kelton, and W. D. Kelton, Simulation modeling and analysis. McGraw-Hill New York, 2000, vol. 3.

[20] P. Palensky, A. A. Van Der Meer, C. D. Lopez, A. Joseph, and K. Pan, “Cosimulation of intelligent power systems: Fundamentals, software architecture, numerics, and coupling,” IEEE Industrial Electronics Magazine, vol. 11, no. 1, 2017, pp. 34–50.

[21] C.-W. Yang and V. Vyatkin, “On automated co-simulation testing of functional requirements for distributed substation automation systems,” in IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society. IEEE, 2018, pp. 3576–3581.

[22] C.-h. Yang, G. Zhabelova, C.-W. Yang, and V. Vyatkin, “Cosimulation environment for event-driven distributed controls of smart grid,” IEEE Transactions on Industrial Informatics, vol. 9, no. 3, 2013, pp. 1423–1435.

[23] F. Schloegl, S. Rohjans, S. Lehnhoff, J. Velasquez, C. Steinbrink, and P. Palensky, “Towards a classification scheme for co-simulation approaches in energy systems,” in 2015 International Symposium on Smart Electric Distribution Systems and Technologies (EDST). IEEE, 2015, pp. 516–521.

[24] M. Büscher, A. Claassen, M. Kube, S. Lehnhoff, K. Piech, S. Rohjans, S. Scherfke, C. Steinbrink, J. Velasquez, F. Tempez et al., “Integrated smart grid simulations for generic automation architectures with rt-lab and mosaik,” in 2014 IEEE International Conference on Smart Grid Communications (SmartGridComm). IEEE, 2014, pp. 194–199.

[25] A. M. Kosek, O. Lünsdorf, S. Scherfke, O. Gehrke, and S. Rohjans, “Evaluation of smart grid control strategies in co-simulation—integration of ipsys and mosaik,” in 2014 Power Systems Computation Conference. IEEE, 2014, pp. 1–7.

[26] C. Steinbrink, M. Blank-Babazadeh, A. El-Ama, S. Holly, B. Lüers, M. Nebel-Wenner, R. P. Ramírez Acosta, T. Raub, J. S. Schwarz, S. Stark et al., “Cpes testing with mosaik: Co-simulation planning, execution and analysis,” Applied Sciences, vol. 9, no. 5, 2019, p. 923.

[27] T. A. Henzinger, “The theory of hybrid automata,” in Verification of digital and hybrid systems. Springer, 2000, pp. 265–292.

[28] K. Conboy, R. Gleasure, and E. Cullina, “Agile design science research,” in International Conference on Design Science Research in Information Systems. Springer, 2015, pp. 168–180.

[29] H. Waltl and H. Wildemann, Modularisierung der Produktion in der Automobilindustrie. TCW, Transfer-Zentrum, 2014.