Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 138 (2026) 773–778

www.elsevier.com/locate/procedia

18th CIRP Conference on Intelligent Computation in Manufacturing Engineering

Assembly System Modeling with SysML and AutomationML for Configuration and Reconfiguration

Jan-Erik Rath[a]*, Johann Gierecker[a], Thorsten Schüppstuhl[a]

[a]*Hamburg University of Technology, Institute of Aircraft Production Technology, Denickestr. 17, 21073 Hamburg, Germany*

* Corresponding author. Tel.: +49-40-42878-4918; fax: +49-40-42731-4551. *E-mail address:* jan-erik.rath@tuhh.de

Abstract

Product individualization and the need to reduce lead times put increasing pressure on production system engineering. This includes the configuration of assembly systems regarding a new product variant. Therefore, model-based engineering is increasingly applied in development and configuration tasks. To combine the two popular practices of using SysML in product development and AutomationML in assembly system development, we propose to link both in an integrated modeling approach. It aims at a seamless assembly system configuration process building upon the capabilities of AutomationML for modeling and interconnecting products, processes, and resources, including their geometries, kinematics, and behaviors.

Keywords: Assembly; Modelling; Planning; Reconfiguration; System

1. Introduction

The trend towards increasingly individualized products can be seen in numerous industries, such as automotive, aviation, and sports equipment manufacturing [1]. For example, seats, overhead stowages, and personal entertainment options such as screens or power outlets are only a few customization options for an aircraft cabin. The wide variety and increasing modularization of the product also demand high flexibility and reconfigurability of the assembly system [2]. Configuration tasks can include the change of assembly sequences, the change, new allocation, or physical reconfiguration of assembly resources such as jigs, tools, or robots, and also more detailed engineering tasks such as programming of control code or the change of a parameter such as the torque in a screwing process [2–4]. These changes or even the completely new planning of an assembly system are often manual tasks conducted by experts. They require deep knowledge of the product as well as the assembly process, the available assembly resources, their capabilities, and, at best, also the previously assembled products. In many cases, product design and assembly planning are carried out separately [5], and the different information is distributed across multiple sources, which may not be up-to-date. Therefore, assembly system planning and configuration tasks are time-consuming and error-prone, leading to long lead times and iterative and costly processes in product design, assembly system design, and assembly [6].

To overcome this, digital models of product and assembly system data are needed that are always up-to-date and available to anyone in the development and configuration process at the necessary level of detail and in a known and established format for the respective task. The utilization of machine-readable models furthermore enables automation or data-based decision-support systems in planning and configuration tasks, allowing an earlier evaluation of change influences between product development and assembly planning [7].

10.1016/j.procir.2026.01.133

Current approaches for model-based co-development and co-configuration are presented in the next section, highlighting the need for a common modeling concept ranging from early product development up to detailed assembly system planning while still enabling engineers to use their discipline-specific modeling tools. Such a concept involving SysML and AutomationML is proposed in section 3 and applied to the example of a smart jig for aircraft cabin pre-assembly in section 4 before the paper closes with a conclusion and an outlook.

2. State of the Art

Current modeling approaches range from Product Data and Product Lifecycle Management Tools, which are dependent on rather inflexible proprietary software platforms, to Model-Based Systems Engineering (MBSE), which is increasingly used as a holistic modeling approach, especially in the early development stages of cyber-physical systems [8]. In this context, the Systems Modeling Language (SysML) enables the traceable modeling of requirements and structural and behavioral information at different hierarchical levels while not eliminating the need for discipline-specific tools and models, posing a challenge to their integration [9]. In the domain of production and assembly system engineering, the Automation Markup Language (AutomationML) was established as an open, XML-based, object-oriented standard for modeling and exchanging plant engineering data [11]. It combines the representation of hierarchies, attributes, relations, and interfaces of objects with geometrical and kinematic information, behavior, and sequencing information, including control code, Sequential Function Charts (SFC), and robot programs [10,11].

Until now, concepts for the co-development or co-customization of products and production or assembly systems mainly stayed within one of the different modeling tools and focused on a specific configuration activity, such as assembly sequence planning. For example, Schäfer et al. [12] automated the generation of production sequences from CAD data with the help of a reference system modeled in SyML. Sierla et al. [13] combined assembly sequence planning with trajectory planning by modeling product interfaces and location parameters in AutomationML. Järvenpää et al. [4] used ontology modeling to implement a capability-based resource allocation process. Similarly, Cavin and Lohse [14] present the concept of decomposing a process requirement into lower-level skill requirements and propose an abstract resource allocation methodology. For the same purpose, products, processes, and resources (PPR) are defined as basic modeling elements in many approaches and also in AutomationML [11]. This well-known division is extended with a skill definition in AutomationML by Pfrommer et al. [15]. Building upon this, Backhaus and Reinhart [16] model products, processes, resources, and skills in AutomationML to enable task-oriented programming of assembly systems. In the aircraft cabin context, Ghanjaoui et al. [5] propose a methodology to link product development and assembly, building upon a PPR system architecture model in SysML combined with knowledge-based assembly process planning.

To summarize, the basic concepts used in many approaches to combining product and assembly system modeling are very similar and build upon abstract PPR and skill model elements. However, depending on the background of the research, a more product-centric or assembly-centric view was taken. While the former often relies on SysML, the latter explores the detail-engineering and exchange capabilities of AutomationML. Thus, in the following, an approach is presented that takes advantage of both modeling languages, ranging from abstract product definition to detailed control code configuration and enabling automated or assisted assembly system planning and reconfiguration.

3. Assembly System Model

3.1. Common metamodel and ontology for product and assembly system modeling

The modeling of assembly-relevant product information in coherence to the assembly system information is a key to enabling assistive or automated software tools for assembly system configuration. To enable a mutual understanding of the two domains while still using the domain-specific modeling languages, a common basic metamodel must be formulated. Berschik et al. [17] proposed such a metamodel in order to connect the lifecycle phases of an aircraft cabin. An adaption of this metamodel is depicted in Fig. 1. It allows a distinguishment between different lifecycle phases, such as product development or assembly planning, on the one hand, and a hierarchical level on which a model is defined, such as enterprise-, plant- or work-cell-level, on the other hand. On each level, model elements of four groups can be used, namely requirement, functional, logical, and physical elements. The data to be modeled is represented by different datatype elements, which consist of a name, a unique identifier, zero to many properties, and references to internal or external data. A datatype can be, for example, a product module, an assembly resource, or an assembly process.

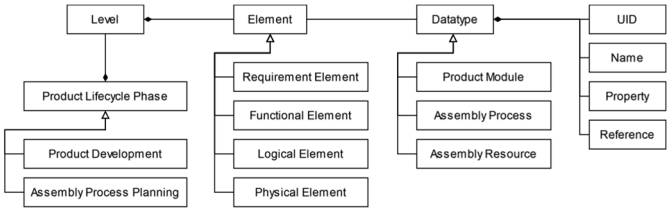

Fig. 1 Common metamodel for product and assembly system modeling

Berschik et al. [17] and Rath et al. [18] further defined an ontology to connect the lifecycle phases of product development and assembly system planning following the PPR paradigm. Its most important relationship is refined and shown in Fig. 2. A product module to assemble has a functional assembly requirement. This high-level requirement may be broken down into multiple assembly processes, which are then realized by one or more resources that provide their skills in the assembly system. The logical order of assembly processes is defined in the assembly sequence. Following the metamodel, the depicted elements are of requirement, functional, logical, or physical-type and can all contain properties and references.

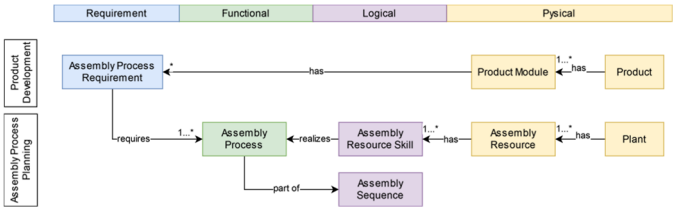

Fig. 2 Ontology linking product development and assembly process planning

3.2. SysML modeling of assembly requirements

As mentioned in the state of the art, model-based engineering of mechatronic products is often supported by SysML to allow traceability between product requirements, functions, logical and physical elements. Therefore, SysML was chosen as a modeling language for the product information in this work. Most relevant for assembly sequence generation, resource allocation, and work order authoring is the information on how the individual product modules are to be connected. Therefore, the DIN 8593 [19] classification of joining operations was chosen for functional modeling of assembly requirements on a high hierarchical level. The information on component connections could be derived from CAD data, but the utilization of SysML also enables traceable modeling of assembly requirements which are derived from product requirements, functions, behavior, and physical information. To enable efficient modeling, all DIN 8593 joining processes can be defined as stereotypes in SysML. A Block Definition Diagram (BDD) is then used to depict the product structure from an assembly viewpoint. Therein, the individual product modules to be assembled are represented as blocks with properties such as mass and bounding box dimensions, as well as references, for example to a CAD file. Directed associations are used to model the relationship between these modules. Thereby, the stereotype library can be used to assign a type of connection as a functional assembly requirement to an association, as shown in Fig. 3. Further information can be added as properties to the association, which also carries the multiplicity of the components.

Fig. 3 BDD of a product module associated with another product module via its assembly requirement formulated as a DIN8593 joining operation

3.3. AutomationML assembly system modeling

As AutomationML is predominant in research for assembly system modeling, it shall be used to model the products, processes, and resources on the assembly system side. In coherence with the general concept of AutomationML, the RoleClassLibrary (RCL) is used to define generic resource and process classes. These predefined classes can be used to model the real available resources and processes of the specific assembly system context, such as the plant, in a Resource and Process SystemUnitClassLibrary (SUCL), respectively. While the RoleClasses are equipped with relevant empty properties and interfaces from the AttributeTypeLibrary and the InterfaceClassLibrary, these attributes and interfaces are filled with specific values in the SUCL. Furthermore, the resources are equipped with their process skills as Internal Elements (IE). From the SUCL, resources and processes can then be instantiated in the Resource and Process InstanceHierarchy (IH) to represent the configured assembly system for a given product configuration, which itself is imported from a SysML model. The different model elements and concepts are shown in Fig. 4 and further detailed in the following subsections.

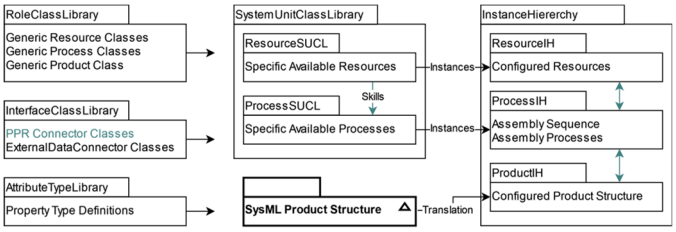

Fig. 4 Overview of the assembly system modeling concept in AutomationML

3.3.1. Modeling of assembly requirements

To match assembly requirements with resource skills and to provide a full representation of the assembly system in AutomationML, the assembly-relevant product information discussed in section 3.2 must be translated from SysML into AutomationML. To enable this, the DIN 8593 joining operations are modeled as generic RoleClasses in the Process RCL. However, the type of joint only defines an assembly requirement on a high hierarchical level, and assembly planners must be able to further detail the assembly process. Therefore, the VDI 2860 classification of handling functions, the DIN 8580 classification of adjustment functions, and additional auxiliary and special operations are additionally defined as Process RoleClasses in the RCL, as shown in Fig. 5a. To support the process planning, in the Process SUCL, each DIN 8593 joining operation is detailed by a sequence of processes which are typically to be realized during the respective joining operation, as InternalElements. Fig. 5b depicts the example of an insertion process, which is further detailed into subfunctions following a Methods-Time-Measurement (MTM) cycle.

Fig. 5 (a) Process RCL with joining processes, handling and additional functions. (b) Process SUCL with joining processes detailed by subfunctions

Translation of a SysML product architecture model into AutomationML is simplified by an XML notation and a common definition of the model artifacts according to the metamodel. As shown in Fig. 6, the SysML product modules (blocks) are instantiated in the AutomationML ProductIH, also adopting the defined references and attributes. The assembly requirements (associations) are instantiated in the ProcessIH. Thereby, due to the common naming convention, the SysML joining operations are matched with the AutomationML joining operations from the ProcessSUCL so that also the predefined sequence of assembly functions within each joining operation can be instantiated or suggested to an engineer for selection or alteration. Similarly, the predefined interfaces of the processes from the SUCL are instantiated. Among them are the handledObject and the targetAssy interfaces, which are connected by InternalLinks to the PPRConnector interfaces of the products to be joined, as shown in Fig. 6.

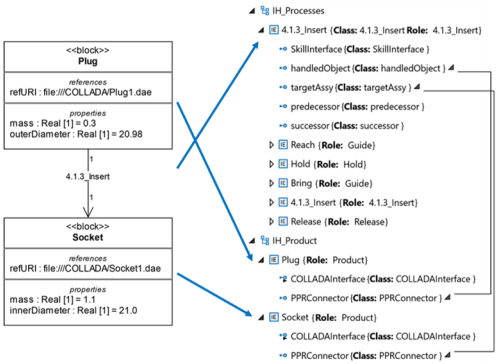

Fig. 6 Insertion of a Plug into a Socket as modelled in SysML (left) and as translated into AutomationML (right)

3.3.2. Modeling of the assembly sequence

One of the most important tasks in assembly system planning is the definition of an assembly sequence. The individual assembly processes can be ordered not only manually, but also by utilizing semantic reasoning and other sequencing algorithms. In AutomationML, an assembly sequence can be modeled by connecting the individual processes' interfaces "predecessor" and "successor" with InternalLinks, as shown in Fig. 7a. This allows the automatic generation of an SFC or other sequence diagram as shown in Fig. 7b, which itself can be referenced on the top level of the ProcessIH.

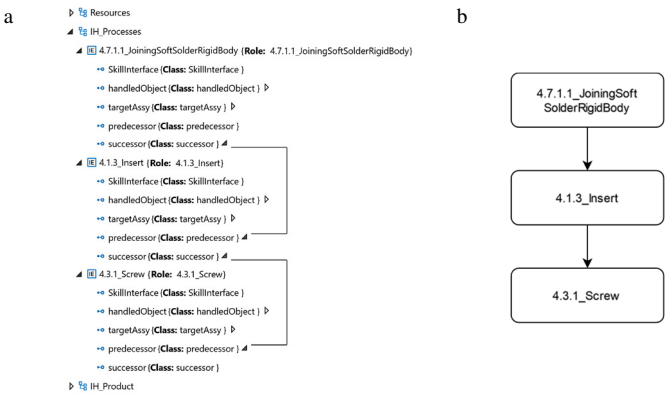

Fig. 7 Example of a simple assembly sequence (a) modelled in AutomationML and (b) visualized as sequence diagram

3.3.3. Resource modeling and resource allocation

Just as the processes, generic assembly resources such as tools, workers, or robots are defined as role classes in the Resource RCL. The specific resources available in the context of the assembly system can then be defined in the Resource SUCL, where properties such as cost, weight, and handling capacity are filled with values. Geometric and kinematic information can be referenced via a COLLADAInterface. Furthermore, the skills of the resource are modeled by instantiating processes from the RCL as InternalElements within the resource. Fig. 8a and b show an example, where a worker with the attributes payload, cost, and language offers the high-level skills PutTogether, Press, and JoiningSoftSolder.

The skill modeling enables an automated or assisted resource allocation process based upon a matching of the skills offered by resources available in the SUCL with the required assembly processes from the IH. Depending upon the desired level of detail, as well as the type of resource and process, this matchmaking can take place on both described hierarchical levels of the assembly processes. During the allocation process, predefined rules comparing the assembly requirement of a product module, including relevant properties such as the weight, with the resource skills, including relevant properties such as the payload, enable automatic reasoning. When a resource is allocated to a process, it is instantiated in the Resource IH, and the PPRConnector of its ProcessIE is associated with the SkillInterface of the respective assembly process by an InternalLink, as shown in Fig. 8c.

Fig. 8 (a) Example of resource modeling in the SUCL, including (b) attributes. (c) Modeling of resource allocations by linking resource skills with processes in the Resource and Process InstanceHierarchies.

3.3.4. Modeling of resource behavior and configuration of assembly instructions

One of the main benefits of AutomationML in the context of assembly system configuration is the possibility of linking the model elements with behavior models, control logic, and robot programs. Therefore, the effort in generating assembly instructions or executable control code during a planning or configuration process can be greatly reduced. While an overall logic of the desired assembly system behavior is built by defining the assembly sequence on the process side, each resource has to be addressed differently to inform it of its desired behavior. While workers require human-readable assembly instructions in their preferred languages, a robot con-

troller needs an interpretable program code. Therefore, the concept of SkillTemplates is introduced. In the SUCL, each resource is linked to an empty shell of a work instruction in a format that it can interpret. This can be for example a textual work instruction template referenced by an ExternalDataConnector or the shell of a PLC program referenced by a PLCopenXMLInterface. Furthermore, each process skill IE offered by a resource in the SUCL is linked to the specific work instruction element able to trigger the desired behavior when added to the SkillTemplate shell of the resource. For certain resources such as grippers, the SkillTemplate can be the inherent behavior of the resource modeled in an IEC 61131-3 conform way such as a State Chart or SFC. The individual process skill is then linked to a specific element of the SFC with a PLCopenXMLInterface, as shown in Fig. 9.

In a software-assisted assembly system planning or configuration process, the configuration of SkillTemplate shells with specific process SkillTemplates takes place when a resource skill gets allocated to a required assembly process. Thereby, the SkillTemplate is instantiated as the actual work instruction for the resource in the InstanceHierarchy. Due to the linkage of product, process, and resource information, an automated parametric configuration of the assembly instructions is enabled. This can include setting torque, speed, or other parameters in a SkillTemplate, especially enabling an efficient reconfiguration of assembly systems after parametric changes.

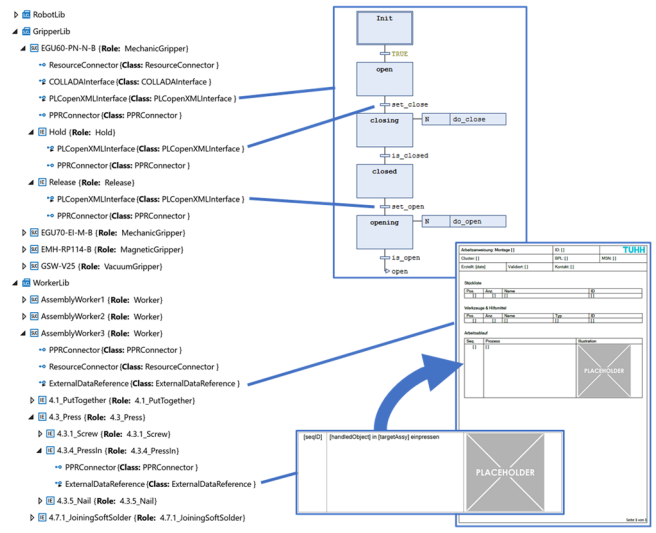

Fig. 9 Two examples of SkillTemplates defined in the SUCL: inherent behavior of a gripper (above) and human-readable work instruction template of a worker with process-specific work instruction element (below)

4. Application Example

A modular smart jig for the assembly of variable aircraft cabin components was chosen as an application example to test the modeling method. The jig is used for the pre-assembly of cabin modules, enabling early testing of components and reducing efforts in final assembly steps. As cabin modules such as overhead stowage compartments (OHSC) vary in dimensions, mass, mounting position, and subassemblies to install, a high number of individual fixtures would be required. Thus, a modular jig that can be flexibly configured for the specific assembly task is beneficial.

As a simple example use case, two product modules are to be pre-assembled, connecting them through a bayonet mount (InsertLock). Fig. 10a shows the product modules and process requirement, imported from a SysML model and instantiated from the AutomationML SUCL in the IH. The predefined sequence of detailed assembly functions must be adapted and extended for the specific process, leading to the assembly sequence depicted in Fig. 10b and modeled in Fig. 10c.

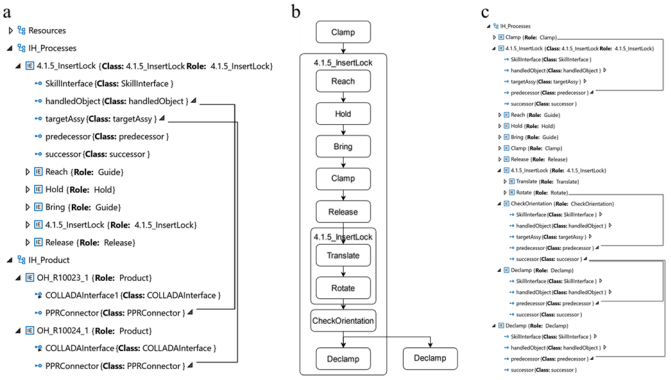

Fig. 10 (a) Product modules and process requirement as imported from a SysML model and instantiated in the IH. (b) Assembly Sequence Diagram. (c) ProcessIH with additional process steps and sequence information. For readability, not all elements and links can be shown.

The main task of the planning process is to configure the jig to fixate and pre-assemble the two modules. To do so, the assembly system planner can rely on multiple jig modules such as main body, height and angular adjustment, linear actuator, fixture, and sensor modules, as well as further resources modeled with their properties and skills in the SUCL. The model of required processes and products with their parameters in the IH enables the engineer and an assistive software tool with underlying matching rules to allocate and configure the different jig modules with limited effort. An excerpt of the SUCL is depicted in Fig. 11, and the configured and allocated resources in the IH in Fig. 12. While one of the product modules is already clamped in the stationary fixture of the jig, the other product module is inserted into the movable fixture by an assembly worker. After the product is clamped, the linear actuator performs a translative motion to connect the two modules. Subsequently, the worker rotates the bayonet mount to fixate the connection. Finally, a sensor integrated into the jig checks the orientation of the bayonet before the product is declamped.

Fig. 11 SystemUnitClassLibraries of the available jigs, fixtures, and measuring devices for the configuration of a modular jig

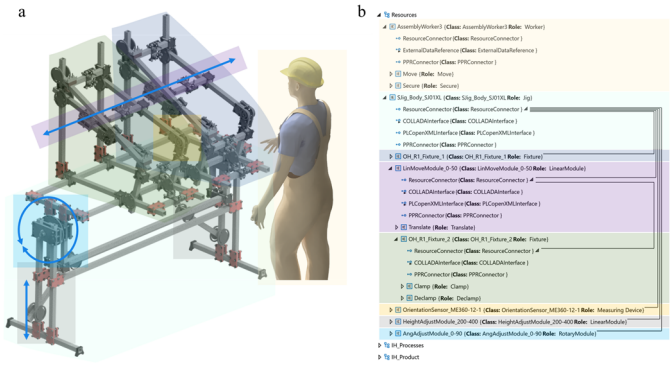

Fig. 12 (a) CAD rendering of the configured modular smart jig and worker during the assembly process. (b) IH of the configured and allocated resources.

5. Conclusion and Outlook

The increasing demand for individualized products across industries requires flexible and reconfigurable assembly systems as well as model-based approaches to facilitate their co-development and co-configuration by automated or decision-support systems. To enable mutual understanding between product development and assembly system planning while still enabling engineers to use their discipline-specific modeling tools, this research introduces an integrated modeling approach using SysML and AutomationML, building upon a common metamodel and ontology for product and assembly system modeling. SysML is utilized to capture product structures and general assembly requirements, while in AutomationML, the assembly processes are detailed and sequenced, the available resources with their skills are described, and the interconnections between products, processes, and resources are modeled. The newly introduced concept of SkillTemplates furthermore enables the assisted or automated configuration of control programs and work instructions.

The application example of a modular smart jig for aircraft cabin pre-assembly demonstrates the practical applicability and effectiveness of the proposed modeling method. Based upon the SysML and AutomationML models, assembly system planners can quickly configure and adapt the jig for specific assembly tasks, reducing manual effort and facilitating virtual commissioning.

Future work can focus on exploring the modeling approach with advanced optimization algorithms and machine learning techniques to automate and optimize assembly system planning and configuration. This will establish a quick feedback generation to evaluate the influence of changes on the overall system. Integration of the proposed modeling approach with Industry 4.0 concepts can further enable real-time monitoring and adaptive reconfiguration of assembly systems, enhancing their resilience.

Acknowledgements

Research was funded by the German Federal Ministry for Economic Affairs and Climate Action under the program LuFo VI-1 Verdika as well as IFB Hamburg under the program GATE prepAIR II.

References

[1] Fogliatto FS, Da Silveira GJ, Borenstein D. The mass customization decade: An updated review of the literature. Int J Prod Econ 2012;138(1):14–25. https://doi.org/10.1016/j.ijpe.2012.03.002.

[2] ElMaraghy HA. Flexible and reconfigurable manufacturing systems paradigms. Int J Flex Manuf Syst 2005;17(4):261–76. https://doi.org/10.1007/s10696-006-9028-7.

[3] Drath R (ed.). Automationml: A practical guide. 1st ed. Boston: De Gruyter Oldenbourg; 2021.

[4] Järvenpää E, Siltala N, Hylli O, Lanz M. Capability Matchmaking Procedure to Support Rapid Configuration and Re-configuration of Production Systems. Procedia Manuf 2017;11:1053–60. https://doi.org/10.1016/j.promfg.2017.07.216.

[5] Ghanjaoui Y, Satwan P, Biedermann J, Nagel B. Model-based Assembly Process Planning for Flexible Aircraft Cabin Architectures; 2024.

[6] Dashchenko OA, Elchov PE, Dashchenko AI. Application of Non-Traditional Assembly Methods in Reconfigurable Manufacturing. In: Dashchenko AI, editor. Reconfigurable Manufacturing Systems and Transformable Factories. Berlin: Springer; 2006, p. 569–581. https://doi.org/10.1007/3-540-29397-3_28.

[7] Schuh G, Potente T, Wesch-Potente C, Weber AR, Prote J-P. Collaboration Mechanisms to Increase Productivity in the Context of Industrie 4.0. Procedia CIRP 2014;19:51–6. https://doi.org/10.1016/j.procir.2014.05.016.

[8] Muggeo C, Pfenning M. Die Rolle von MBSE und PLM im Industrial Internet. In: Schulze S-O, Muggeo C, Abulawi J, editors. Tag des Systems Engineering. München: Hanser; 2016, p. 279–287. https://doi.org/10.3139/9783446447288.028.

[9] Alt O. Modellbasierte Systementwicklung mit SysML. München: Carl Hanser Fachbuchverlag; 2012.

[10] Stark K, Goldschmidt T, Doppelhammer J, Bihani P, Goltz D. Cloud-based integration of robot engineering data using AutomationML. In: 14th International Conference on Automation Science and Engineering (CASE). IEEE; 2018.

[11] Drath R. Datenaustausch in der Anlagenplanung mit AutomationML. Berlin, Heidelberg: Springer; 2010.

[12] Schäfer L, Frank A, May MC, Lanza G. Automated Derivation of Optimal Production Sequences from Product Data. Procedia CIRP 2022;107:469–74. https://doi.org/10.1016/j.procir.2022.05.010.

[13] Sierla S, Kyrki V, Aarnio P, Vyatkin V. Automatic assembly planning based on digital product descriptions. Comput Ind 2018;97:34–46. https://doi.org/10.1016/j.compind.2018.01.013.

[14] Cavin S, Lohse N. Multi-level skill-based allocation methodology for evolvable assembly systems. In: 2014 12th IEEE International Conference on Industrial Informatics (INDIN). IEEE; 2014, p. 532–537. https://doi.org/10.1109/INDIN.2014.6945569.

[15] Pfrommer J, Schleipen M, Beyerer J. PPRS: Production skills and their relation to product, process, and resource. In: 2013 IEEE 18th Conference on Emerging Technologies & Factory Automation (ETFA). IEEE; 2013. https://doi.org/10.1109/ETFA.2013.6648114.

[16] Backhaus J, Reinhart G. Digital description of products, processes and resources for task-oriented programming of assembly systems. J Intell Manuf 2017;28(8):1787–800. https://doi.org/10.1007/s10845-015-1063-3.

[17] Berschik MC, Blecken M, Kumawat H, Rath J-E, Krause D, God R et al. A holistic aircraft cabin metamodel as an approach towards an interconnected digitised cabin lifecycle. In: 33rd Congress of the International Council of the Aeronautical Sciences, ICAS 2022; 2022. https://doi.org/10.15480/882.4757.

[18] Rath J-E, Koch J, Schüppstuhl T. Towards Model-Based Assembly System Configuration Supported by SysML and AutomationML. In: Silva FJG, Pereira AB, Campilho RDSG, editors. Flexible Automation and Intelligent Manufacturing: Establishing Bridges for More Sustainable Manufacturing Systems. Cham: Springer Nature Switzerland; 2024, p. 622–632. https://doi.org/10.1007/978-3-031-38241-3_70.

[19] DIN 8593-0:2003-09, Fertigungsverfahren Fügen_- Teil_0: Allgemeines; Einordnung, Unterteilung, Begriffe. Berlin: Beuth Verlag GmbH; 2003. https://doi.org/10.31030/9500684.