5th Conference on Production Systems and Logistics

Interoperable Architecture For Logical Reconfigurations Of Modular Production Systems

Sebastian Behrendt[1], Marco Wurster[1], Matthias Strljic[2], Jan-Felix Klein[3], Marvin Carl May[1], Gisela Lanza[1]

[1]*wbk Institute of Production Science, Karlsruhe Institute of Technology*
[2]*Institute for Control Engineering of Machine Tools and Manufacturing Units, University of Stuttgart*
[3]*Institue for Material Handling and Logistics, Karlsruhe Institute of Technology*

Abstract

Individualisation of products and ever-shorter product lifecycles require manufacturing companies to quickly reconfigure their production and adapt to changing requirements. While most of the existing literature focuses on organisational structures or hardware requirements for reconfigurability, requirements and best practices for logical reconfigurations of automated production systems are only sparsely covered. In practice, logical system reconfigurations require adjustments to the software, which is often done manually by experts. With the ongoing automation and digitisation of manufacturing systems in the context of Industry4.0, the need for automated software reconfigurations is increasing. However, heterogeneous and proprietary technologies in the field of industrial automation pose a hurdle to overcome for generally applicable approaches for logical reconfigurations in the industrial domain. Therefore, this paper reviews available technologies that can be used to solve the problem of automated software reconfigurations. For this purpose, an architecture and a procedure are proposed on how to use these technologies for automatic adaptation and virtual commissioning of control software in industrial automation. To demonstrate the interoperability of the approach, collective cloud manufacturing is used as a composing platform. The presented approach further includes a domain-specific capability model for the specification of software artefacts to be generated, allowing jobs to be described and matched on the platform. The core element is a code generator for generating and orchestrating the control code for process execution using the reconfigurable digital twin as a validator on the platform. The approach is evaluated and demonstrated in a real-world use case of a modular disassembly station.

Keywords

Logical Reconfiguration; Production Control; Industry4.0; Modular Production Systems; Reconfigurable Manufacturing Systems

1. Introduction

Today's rapid and dynamic business environment, characterised by trends like individualised products and shorter product life cycles, is forcing manufacturing companies to adapt their production systems frequently and efficiently [1, 2]. In addition, volatile and uncertain material supply and customer demand [3] require the deployment of adaptive production systems that can swiftly respond to changing circumstances. Reconfigurability proves to be a key capability to enable manufacturers to provide production processes and resources on demand [4].

DOI: https://doi.org/10.15488/15270
ISSN: 2701-6277

Industry4.0 and Cloud Manufacturing (CM) are driving the integration of cyber-physical systems (CPS) into the production environment to create cyber-physical production systems (CPPS) [5]. The seamless convergence of software and hardware forms the backbone of adaptive production systems, with software becoming an integral part [6]. Although the extension of hardware with software comes at the cost of increased complexity, it also enables digital planning, deployment and control of manufacturing operations [7]. The Software-defined Manufacturing (SDM) paradigm promises to enable orchestration and management of large CPS by using abstraction and encapsulation to reduce complexity when interacting with heterogeneous hardware and software infrastructure [8, 9].

While there exist many frameworks that provide design guidelines for software and hardware of flexible and reconfigurable production systems, their concrete implementation raises many questions given the wide range of technologies used in industrial practice. Reference architectures for CPS, such as RAMI4.0 or IIRA, provide conceptual and abstract frameworks but their usefulness for realisation of CPPS is limited due to their lack of technical depth, according to Wang et al. [10]. In fact, changing the logic of CPPS by reconfiguring software, e.g. by changing sequencing and parametrisation of control code, is still a highly manual task in practice, which increases cost of reconfigurations and limits the useable potential of reconfigurable hardware [11–13].

Heterogeneity of communication protocols, proprietary software interfaces and complicated planning tasks challenge the general applicability of more technical approaches for logical reconfigurations, especially with respect to the integration and automation of software systems [10]. In the following, we will explore existing approaches that contribute the reconfigurability of production systems and evaluate their suitability (Section 2). With reference to ElMaraghy [1], Wiendahl et al. [4] and Monostori [5], we propose the following requirements to evaluate existing approaches for realising reconfigurable production systems:

R1 Automation: Automated adaptation of the control code to suit individual hardware configurations.

R2 Abstraction: Individual software modules are encapsulated according to their associated hardware, abstracting the hardware and its capabilities in a service-oriented manner.

R3 Generalisation: The architecture is generic and can be implemented with various tools and technologies, thus promoting interoperability.

R4 End-to-End: Continuous approach from the gathering of the customer requirements to the physical execution of processes.

R5 Virtual Validation: The feasibility of the sequenced processes should already be predicted virtually with a high quality.

R6 Integration: The integration of additional hardware/software modules and their deployment in the existing infrastructure should be possible with little effort.

Based on the findings of this analysis, we present a framework and an architecture that uses state-of-the-art technologies and the concepts of CM and SDM to overcome existing limitations (Section 3). Section 4 demonstrates and validates the applicability of this approach in a case study of a reconfigurable and automated assembly station. Finally, Section 5 discusses the potentials and limitations of the proposed approach and gives directions for research in the field of reconfigurable production systems.

2. State-of-the-Art

Several approaches aim to provide a framework for designing production systems that can better adapt to changing requirements through flexibility and changeability [4]. While both flexibility and changeability specify the ability to adjust the operating point of production systems, changeability goes beyond flexibility by covering more fundamental changes to the production system, such as reconfigurations [14].

Existing approaches in the literature for manufacturing system paradigms have in common that they aim to have universal production resources that can be equipped with different processes, thus allowing simple reconfigurations of the production capacity. Bionic, fractal and holonic manufacturing systems [15, 16] follow the idea of having universal entities that can act autonomously, change their configuration and interact with each other to enable an adaptive system. Another paradigm, i.e. reconfigurable manufacturing systems (RMS), provides a more tangible approach by providing design principles for reconfigurability [17]. Reconfigurability specifies the ability to adapt the physical or logical structure to implement a different functionality, both at the resource level and at the system level [18].

More recent paradigms, in particular Smart Manufacturing Systems and Advanced Manufacturing Systems, focus more on the IT infrastructure required for logical reconfigurations, but the literature lacks generally applicable approaches for implementing these paradigms in practice [10, 19, 20]. Existing approaches use proprietary solutions that are not generally applicable due to technical barriers such as different programmable logic controller (PLC) manufacturers or proprietary protocols. For example, Talkhestani et al. [12] adapt PLC control code for logical reconfigurations by using a direct but proprietary PLC interface.

Besides the field of manufacturing system's paradigms, there are many approaches in the field of smart manufacturing, Industry4.0 and CM that present concepts and architectures to manage and operate CPS. The integration of generated software artefacts is a core element of model-based systems engineering and has been striving for generated and reusable software within automation technology since the 1990s, as presented by Eppinger and Steven [21]. The concept is being revived in current topics such as the asset administration shell of the reference architecture model RAMI4.0 [22] and the concept of SDM by Verl et al. [23] for autonomous program parameterisation. As described by Wymore [24], the concept of generated control code is used in systems engineering and could be applied more broadly in automated capability-based systems to generate digital twins for process validation, according to Madni et al. [25]. Exemplary approaches already exist, such as a cloud-based evaluation platform for real-time applications within SDM [26]. However, there is still a lack of software tools that validate and generate control code regarding a uniform process description in a neutral process model. For example, Brovkina [27] specifies a process model with a focus on assembly sequences, but this is part of a holistic tool for automated layout design and control code generation with a clearly specified methodology. For example, several studies on general [28], additive [29] and CNC (Computerised Numerical Control) [30] CM platforms show that manufacturing processes for individual components and assemblies can be realised. However, there is a limitation to geometrical and material-related properties that can be covered. The lack of uniform task and process descriptions as well as generally applicable tools for control code generation hinders to provide control code as a service to allow manufacturing machines to be operated adaptively with frequent logical reconfigurations. Industrial architecture models in CM provide metamodels for domain-specific modelling of digital twins to approximate the production capabilities of a service provider [31]. Yet, do not provide additional tools to realise these capabilities by generation of associated control code.

Considering related research in the field of virtual commissioning [32], Martinez et al. [33] present a digital twin demonstrator that adapts the control code based on production orders and resource configurations. After a successful adaptation phase, a digital twin is used to validate the system by simulation.

In the area of process sequence orchestration, Pfrommer et al. [34] present a skill-based framework for reconfigurable manufacturing systems. Here, executable actions can be generated from a high-level description of the capabilities, which are orchestrated as services in the next step, similar to the approach of Backhaus und Reinhart [35]. As noted by van de Ginste et al. [36], the transformation of high-level skills to executable skills remains a research gap. First approaches to automate code generation and constraint checking are provided by Kocher et al. [37].

In summary, there exist many approaches that aim to increase the flexibility and reconfigurability of production systems by considering new design paradigms, especially with regard to CPS and CM. However, there seems to be no approach in the literature that satisfies all requirements presented in section 1. The analysis showed that approaches from the field of manufacturing systems paradigms and reference architectures for digital manufacturing promote Abstraction (R2) and Generalisation (R3) but lack details considering Automation (R1) and Integration (R6). More technical approaches, on the other hand, demonstrate Automation (R1) and Integration (R6) but lack general applicability (R3) and their continuity (R4). In the following, we will present a framework and architectures that aim to combine insights from the reviewed literature to satisfy requirements R1-R6 and to enable automatic deployment of logical configurations in production systems.

3. Approach

The resulting approach consists of two subsequent steps: the development of a code generator (CG) and its embedding in the architecture and system model of the Collective Cloud Manufacturing (CCM). The embedding into the CCM is explained first, followed by the structure and functionality of the CG. Figure 1 serves as a high-level architecture and flow diagram, with numerical steps within the CCM platform and alphabetical steps for domain-specific steps of code generation and validation.

For CCM, the two required components are a domain-specific capability model and an encapsulating analysis container of integration patterns [38]. Thus, individual process steps and system capabilities can be freely defined according to Brovkina's process model [27] and receive the design guideline to be enriched with a standardised position node via a start and end pose. A sequence of process steps is realised via a concatenation of such process steps via a "next" relation. For example, this could represent a sequence of handling tasks from a robot where the start pose specifies the pickup of a workpiece and the end pose specifies the target of the handling task itself. In this context, handling tasks are non-value-adding processes that do not need to be noted explicitly for higher abstraction. Relevant process step parameters can be attached directly to a process step node or as sub-elements via their own relationships. The definition of the domain-specific process steps with a process model forms the basis of the execution flow (step 1 - step 10) of the CCM platform.

For individual processes or sub-process, their configured CG are encapsulated as a simulation service according to Strljic and Riedel [38] within an analysis container. This is annotated with the previously defined domain-specific process step parameters, i.e. the process step model. The process steps represent in their union the entire bundle of required manufacturing capabilities to be executed. The annotated analysis container is registered on the CCM platform in step 1).

Using previously created process step models, users can specify individual production process requirements to be realised by a plant, a manufacturing system or a production resource in step 2). According to the CCM-concept, these can be own plants or a mixture of service providers. The requirements are summarised by created digital assets for each step in the process sequence, which are made available downstream to the CG.

In step 3), a completed requirement description can be uploaded to the CCM platform as a project request, and in step 4), the matching process is initiated with the help of a subgraph search for compatible CGs, as described in [39]. The most suitable candidate of the matching process is instantiated on the CCM platform and starts with step 5): the analysis of digital assets for their requirements of manufacturing capabilities. For this purpose, interfaces from the CCM platform are made available to the CG to provide information about the requirements responsible for matching a process step to a production resource. The core of the instantiation is the execution of the CG as well as providing requirements.

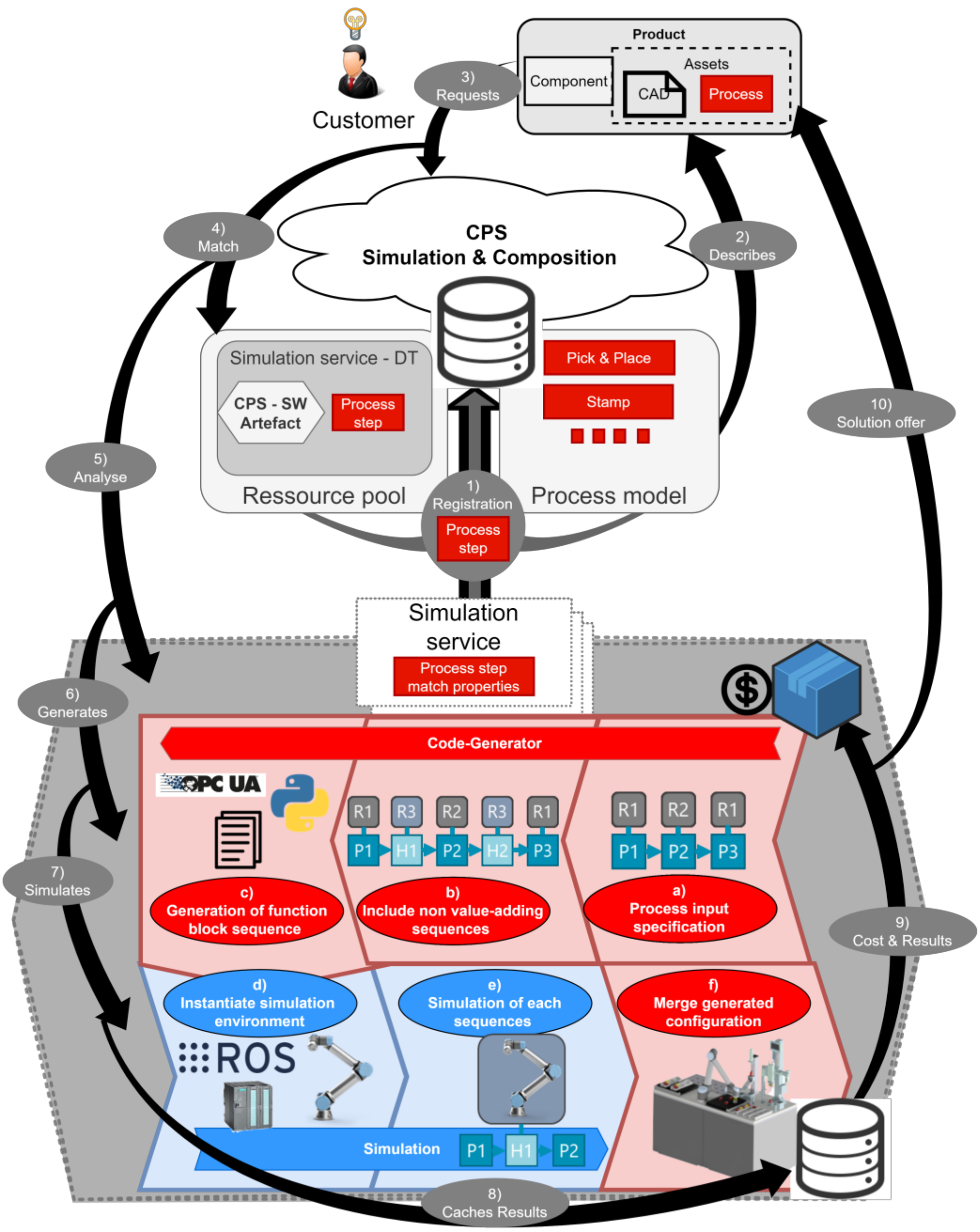

Figure 1 Overview of the architecture for platform-based planning of reconfigurations of production systems with functional sequencing and automated deployment

The CG (step 6) is responsible for translating the process model, which describes the production process sequences in an abstract way, into instructions that are understandable for automated manufacturing equipment. The information that needs to be provided by the CCM platform to the CG is shown in Table 1. This includes resource-related information concerning available production resources and their capabilities and process-related information, i.e. the process sequence to be executed. For a more specific view of the problem of generating and validating control code for RMS, we have subdivided steps 6) and 7) into six more detailed steps a)-f).

With regard to requirement R6 (Integration), the interfaces of the CG used to receive information shall be interoperable and provide self-describing capabilities for efficient integration. The CG should be modular and extensible to accommodate new functionality. To realise this requirement, external data can be passed to the CG (step a) on multiple interfaces, which all get integrated to the internal data model of the CG by data adapters.

Table 1: Overview over input information required to execute the CG classified by the information categories

Information category	**Information content**	**Explanation**
Resource-related information	Communication interfaces	Type and address of communication interfaces
Resource-related information	Configuration	Available processes at a resource and meta information (e.g. positions of process modules)
Resource-related information	Capabilities	Capabilities describe in an abstract way the processes that can be executed by the resource.
Process-related information	Process model	Sequence of processes to execute with required resources and capabilities for these processes.

The control code generation begins with an analysis of whether the provided process sequence in the process model is complete, i.e. all transitional conditions between consecutive process steps are logically valid. For instance, an incomplete process sequence could be missing a process step for handling between two process steps. In the case of missing process steps, the CG determines required state transitions to satisfy the transitional conditions and evaluates whether a suiting resource is available. If a resource is available, the process model is adapted by adding a process step (step b). Otherwise, an error is returned to the CCM platform specifying the missing capability for the transition. In code generation (step c), the CG iterates over the process model and generates code for each process step that can be deployed to the associated production resource. Based on the communication interface and the configuration of the resources, it can be determined, how and what information needs to be provided to adjust the control code. Ideally, code for these processes is pre-planned, deployed, and requires only adjustments without further need for planning and generating new control code. Yet, other processes, such as handling tasks, may need to be planned in the CG. This is achieved by providing interfaces to planning solutions, such as ROS path planning.

Quality of planning can be improved by using simulations. Modelling the system in detail without a high degree of abstraction allows to detect errors in the planning process before the physical commissioning. This reduces additional the effort required during ramp-up of the newly configured station is reduced. The information contained in the process model, the generated control code, and machine-specific information serve as the basis for instantiating simulation environments in step d). Each defined process step describes a change of state of the system starting from an initial state before execution and ending in a final state after execution. The definition of these states allows simulation of individual process steps and their simulative validation (step e). If the final state of the process differs from the planned finals state by a defined tolerance, e.g. due to collisions, the process must be re-planned.

After code generation and validation of each process step, the generated code is merged into a configuration file that is returned to the CCM platform (step f). The file contains the newly planned logical configuration of the production resources with information on how to deploy this configuration to the production resources.

Simulatively determined KPIs of the individual machine modules of the configuration are aggregated and combined within step 9) into a normalised cost model according to production duration. The subordinate cost model is initially only required for ordering and evaluating the planned results, but it can map the concrete economic efficiency of offered process sequences in the broader spectrum. Finally, in step 10), the aggregated KPIs are combined with the resulting control code of the CG into an overall planning result. The result can be imported into an execution instance of the selected CG by a user to run the actual production system with the generated control code for the specified process steps of the process model.

4. Case Study

In this paper, the proposed approach including the CG is exemplarily tested on a modular production system for disassembling electric window lifter motors. The system is part of the AgiProbot factory [40] and built based on the Fluid Automation Design Framework [41]. The latter describes a Plug&Produce architecture for production systems for both hardware and software with the underlying vision to enable fine-granular reconfigruations of the system to follow ever-changing requirements on manufacturers in dynamic and volatile environments [40]. The system consists of multiple reconfigurable stations arranged in a matrix layout. Each station comprises of multiple station modules, which can carry up to four elementary units each. Each elementary unit fulfils a certain function.

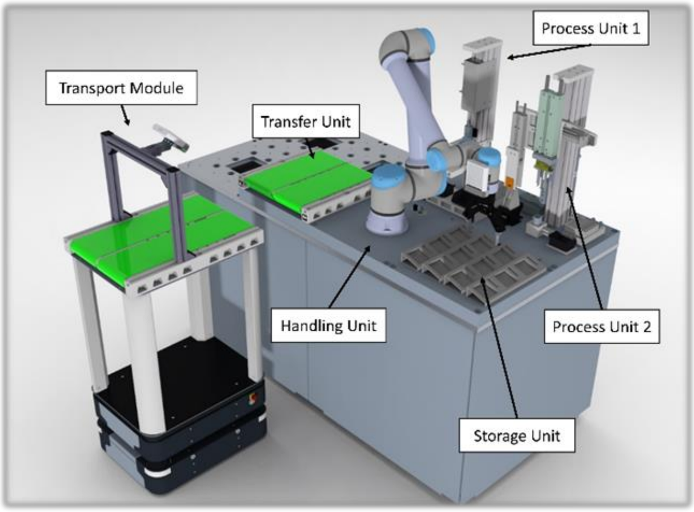

Figure 2 Modular hardware demonstrator based on the fluid automation framework: electric motor disassembly

The disassembly station, as depicted in Figure 2, consists of a handling unit, a storage unit and two process units, which both execute certain tasks of an electric motor's disassembly. A transfer unit moves individual products and components from the station to the transport module, i.e. an autonomous mobile robot. A PLC with an OPC UA interface is responsible to control the processes of individual process units. The handling unit is a universal robot with an interface to the PLC and ROS. Due to its modular design, the station allows to easily reconfigure by changing its process units. To monitor the configuration of process units, i.e. identity, location and rotation of equipped process units, each unit is equipped with RFID tags that specify the process provided by the unit. All resource related information concerning current configuration and meta information such as CAD files can be accessed via the OPC UA server of the station to integrate the station in the CCM platform. The logic of the station is reconfigurable by preplanning and encapsulating control code of each process unit in a function. All functions, each associated with a process step of a process unit, are linked to a specific OPC UA bit. By including a conditional call of every function in the main-loop depending on the associated OPC UA bit, processes can be performed in an arbitrary sequence by triggering their associated OPC UA variable. Moreover, all parameters of the processes can also be adjusted with OPC UA.

The CCM platform provides all information to the CG, comprising of resource and process related information, via REST. The data is parsed into the CG's internal data model (step 6a), and further required process steps are added to the process model based on analysis of location and orientation of the product during the process sequence (step 6b). Code generation (step 6c) for production processes is for the disassembly station trivial, since only the associated OPC UA bits must be specified as control code for every process unit is pre-planned. However, path planning of the handling unit must be done.

Planning and code generation for the handling unit is of particular importance as its kinematic restrictions have a direct influence on the possible configurations of the station. Therefore, the physical 3D simulation environment NVIDIA Isaac in combination with ROS is used as a simulation environment. The process is initiated by transferring all required information from the CG to the planning solution via a ROS action server. This contains the overview of all used assets as well as the initial and final state of this handling step and meta information for simulation and planning. Based on this information, the simulation scenario is instantiated. At first, a fast planning routine is executed, which does not have the goal of finding the best possible path, but to make a statement about the general existence of a valid path. In parallel, optimising algorithms can be used to search for the best possible path. Determined plans are stored in the database and selected plans are additionally tested in the physical simulation. Control code of the planned path is then generated by ROS and returned to the CG.

After code generation and validation, the data required for deploying the new logical configuration to the disassembly station is generated (step 8). This covers a sequence of processes, including process information on how to communicate with the disassembly station for process execution. Subsequently, the data is returned to the CCM platform and can be utilised by an execution client that communicates the information to execute processes to the assembly station via ROS or OPC UA.

5. Discussion and Conclusion

The resulting approach attempts to fully satisfy the requirements R1 - R6, which is done by providing an architecture and a flow diagram for reconfigurable production systems and applying it in a real use case. A container-based system is presented in which the architecture of a CG could be encapsulated for planning and execution of production processes. It is validated in a CM environment to illustrate the interoperability of such an approach. The ability to perform all steps of the flow diagram fully automated in a Docker environment starting from an abstract process description and ending in executable process steps satisfies requirement R1 (Automation) and provides the basis for R4 (End-to-End). By applying the approach to a real-world example, R3 (Generalisation) is also validated. In addition, the interlocking of an abstracted process model with domain-specific CGs shows a high degree of reusability and interchangeability, since domain-specific value-adding process steps can be realised by different underlying hardware, which in turn can be realised by an interchangeable CG. Furthermore, with the integration of interfaces for planning solutions with simulation environments, each process step can be used in a service-oriented manner by a digital twin in its execution for R5 (Virtual validation). The combination of these features, thus, supports R2 (Abstraction). Thanks to the flexible process model and integration of CGs by standardised interfaces of the CCM platform, continuous extensibility to new hardware components is possible through integration via own CGs, process steps or simulation endpoints. Chained planning through multiple CGs allows highly flexible and modular systems to be integrated into more complex projects, thus, achieving an even higher level of abstraction. However, the most crucial limitation of the underlying approach is that it requires a fully digitised production system with open interfaces to control and parametrise processes and evaluate the current configuration of the production system. In addition, CGs with interfaces conforming the interfaces of the CCM platform are required to generate the control code in case of reconfiguration. Since CGs are not widely used yet in practice, this requirement could require a lot of work and adaption of current control infrastructure. Therefore, the presented solution is most efficient in greenfield projects that allow a thorough definition of the software architecture and ideally allow reuse of existing CGs. Yet, the requirements concerning digitisation and availability of open interfaces and CGs prevents application of this approach in most existing production systems. Thus, future research should concentrate on more generally applicable CGs for automated production systems and how these requirements can be reduced to increase the applicability of this approach. Another limitation of the approach becomes present in very complex environments that require execution and orchestration of parallel or concurrent processes. Future research will investigate how this limitation can be overcome and how the approach can be more easily integrated in existing software architectures based on a service-oriented design.

Acknowledgements

The authors thank the Ministry of Science, Research and Arts of the Federal State of Baden-Württemberg for the financial support of the project within the Innovation Campus Future Mobility (ICM).

References

[1] H. A. ElMaraghy, “Flexible and reconfigurable manufacturing systems paradigms,” International Journal of Flexible Manufacturing Systems, vol. 17, no. 4, pp. 261–276, 2006, doi: 10.1007/s10696-006-9028-7.

[2] N. Bennett and G. J. Lemoine, “What a difference a word makes: Understanding threats to performance in a VUCA world,” Business Horizons, vol. 57, no. 3, pp. 311–317, 2014, doi: 10.1016/j.bushor.2014.01.001.

[3] O. Mack, A. Khare, A. Krämer, and T. Burgartz, Eds., Managing in a VUCA world. Cham, Heidelberg, New York, Dordrecht, London: Springer, 2016.

[4] H.-P. Wiendahl et al., “Changeable Manufacturing - Classification, Design and Operation,” CIRP Annals, vol. 56, no. 2, pp. 783–809, 2007.

[5] L. Monostori, “Cyber-physical Production Systems: Roots, Expectations and R&D Challenges,” Procedia CIRP, vol. 17, pp. 9–13, 2014, doi: 10.1016/j.procir.2014.03.115.

[6] R. Baheti and H. Gill, “Cyber-physical systems,” The impact of control technology, vol. 12, no. 1, pp. 161–166, 2011.

[7] X. Xu, Y. Lu, B. Vogel-Heuser, and L. Wang, “Industry 4.0 and Industry 5.0—Inception, conception and perception,” Journal of Manufacturing Systems, vol. 61, pp. 530–535, 2021, doi: 10.1016/j.jmsy.2021.10.006.

[8] A. Lechler and A. Verl, “Software Defined Manufacturing Extends Cloud-Based Control,” in Proceedings of the ASME 12th International Manufacturing Science and Engineering Conference, Los Angeles, California, USA, 2017.

[9] S. Behrendt et al., “Improving Production System Flexibility and Changeability Through Software-Defined Manufacturing,” in Production at the Leading Edge of Technology – Proceedings of the 12th Congress of the German Academic Association for Production Technology (WGP), University of Stuttgart, October 2022. Ed.: M. Liewald, Liewald, Mathiasand Verl, Alexanderand Bauernhansl, Thomasand Möhring, Hans-Christian, Ed.: Springer International Publishing, 2023, pp. 705–716.

[10] B. Wang, F. Tao, X. Fang, C. Liu, Y. Liu, and T. Freiheit, “Smart Manufacturing and Intelligent Manufacturing: A Comparative Review,” Engineering, vol. 7, no. 6, pp. 738–757, 2021, doi: 10.1016/j.eng.2020.07.017.

[11] M. Löppenberg and A. Schwung, “Self Optimisation and Automatic Code Generation by Evolutionary Algorithms in PLC based Controlling Processes,” Apr. 2023. [Online]. Available: https://arxiv.org/pdf/2304.05638.pdf

[12] B. Ashtari Talkhestani et al., “An architecture of an Intelligent Digital Twin in a Cyber-Physical Production System,” at - Automatisierungstechnik, vol. 67, no. 9, pp. 762–782, 2019, doi: 10.1515/auto-2019-0039.

[13] F. Biesinger, D. Meike, B. Kraß, and M. Weyrich, “A digital twin for production planning based on cyber-physical systems: A Case Study for a Cyber-Physical System-Based Creation of a Digital Twin,” Procedia CIRP, vol. 79, pp. 355–360, 2019, doi: 10.1016/j.procir.2019.02.087.

[14] H. ElMaraghy and H.-P. Wiendahl, “Changeable Manufacturing,” in Springer reference, CIRP encyclopedia of production engineering: With 85 tables, L. Laperrière, G. Reinhart, S. Chatti, and T. Tolio, Eds., Berlin: Springer, 2014, pp. 157–163.

[15] A. Tharumarajah, “Comparison of the bionic, fractal and holonic manufacturing system concepts,” International Journal of Computer Integrated Manufacturing, vol. 9, no. 3, pp. 217–226, 1996, doi: 10.1080/095119296131670.

[16] K. Ueda, J. Vaario, and K. Ohkura, “Modelling of Biological Manufacturing Systems for Dynamic Reconfiguration,” CIRP Annals, vol. 46, no. 1, pp. 343–346, 1997, doi: 10.1016/S0007-8506(07)60839-7.

[17] Y. Koren et al., “Reconfigurable Manufacturing Systems,” CIRP Annals, vol. 48, no. 2, pp. 527–540, 1999, doi: 10.1016/S0007-8506(07)63232-6.

[18] Z. M. Bi, S. Y. T. Lang, W. Shen, and L. Wang, “Reconfigurable manufacturing systems: the state of the art,” International Journal of Production Research, vol. 46, no. 4, pp. 967–992, 2008, doi: 10.1080/00207540600905646.

[19] H. ElMaraghy, L. Monostori, G. Schuh, and W. ElMaraghy, “Evolution and future of manufacturing systems,” CIRP Annals, vol. 70, no. 2, pp. 635–658, 2021, doi: 10.1016/j.cirp.2021.05.008.

[20] R. Cioffi, M. Travaglioni, G. Piscitelli, A. Petrillo, and A. Parmentola, “Smart Manufacturing Systems and Applied Industrial Technologies for a Sustainable Industry: A Systematic Literature Review,” Applied Sciences, vol. 10, no. 8, p. 2897, 2020, doi: 10.3390/app10082897.

[21] S. D. Eppinger, “Model-based Approaches to Managing Concurrent Engineering,” Journal of Engineering Design, vol. 2, no. 4, pp. 283–290, 1991, doi: 10.1080/09544829108901686.

[22] Reference architecture model industrie 4.0 (rami4. 0), 2015.

[23] A. Lechler and A. Verl, “Software Defined Manufacturing Extends Cloud-Based Control,” ASME 2017 12th International Manufacturing Science and Engineering Conference collocated with the JSME/ASME 2017 6th International Conference on Materials and Processing, 2017, doi: 10.1115/MSEC2017-2656.

[24] A. W. Wymore, Model-Based Systems Engineering. Boca Raton: Chapman and Hall/CRC, 2018.

[25] A. M. Madni and M. Sievers, “Model-based systems engineering: Motivation, current status, and research opportunities,” Syst Eng, vol. 21, no. 3, pp. 172–190, 2018, doi: 10.1002/sys.21438.

[26] M. Neubauer, C. Ellwein, and F. Frick, “Kontinuität als neues Paradigma,” Computer Automation, vol. 2022, 2022.

[27] D. Brovkina and O. Riedel, “Assembly Process Model for Automated Assembly Line Design,” in 2021 IEEE 3rd Eurasia Conference on IOT, Communication and Engineering (ECICE), 2021.

[28] Strljic, Matthias Milan and Younes, Islam and Riedel, Oliver, Ed., Study on Available Cloud Manufacturing Platforms for Additive Manufacturing Technologies: IEEE, 2022.

[29] M. M. Strljic, I. Younes, and O. Riedel, “Study on Available Cloud Manufacturing Platforms for Additive Manufacturing Technologies,” in 2022 IEEE 4th Eurasia Conference on IOT, Communication and Engineering (ECICE), 2022.

[30] M. M. Strljic, A. Bakovic, and O. Riedel, “Study on Available Cloud Manufacturing Platforms for CNC Milling Processes,” in Study on Available Cloud Manufacturing Platforms for Additive Manufacturing Technologies, 2022.

[31] M. M. Strljic, D. Brovkina, T. Korb, O. Riedel, and A. Lechler, “Platform architecture concept for the composition of collective cloud manufacturing,” in 2018 25th International Conference on Mechatronics and Machine Vision in Practice (M2VIP), 2018.

[32] T. Lechler, E. Fischer, M. Metzner, A. Mayr, and J. Franke, “Virtual Commissioning – Scientific review and exploratory use cases in advanced production systems,” Procedia CIRP, vol. 81, pp. 1125–1130, 2019, doi: 10.1016/j.procir.2019.03.278.

[33] S. Martinez et al., “A Digital Twin Demonstrator to enable flexible manufacturing with robotics: a process supervision case study,” Production & Manufacturing Research, vol. 9, no. 1, pp. 140–156, 2021, doi: 10.1080/21693277.2021.1964405.

[34] J. Pfrommer, D. Stogl, K. Aleksandrov, S. Escaida Navarro, B. Hein, and J. Beyerer, “Plug & produce by modelling skills and service-oriented orchestration of reconfigurable manufacturing systems,” at - Automatisierungstechnik, vol. 63, no. 10, pp. 790–800, 2015, doi: 10.1515/auto-2014-1157.

[35] J. Backhaus and G. Reinhart, “Digital description of products, processes and resources for task-oriented programming of assembly systems,” (in En;en), J Intell Manuf, vol. 28, no. 8, pp. 1787–1800, 2017, doi: 10.1007/s10845-015-1063-3.

[36] L. van de Ginste, A. de Cock, A. van Alboom, S. Huysentruyt, E.-H. Aghezzaf, and J. Cottyn, “A formal skill model to enable reconfigurable assembly systems,” International Journal of Production Research, pp. 1–16, 2022, doi: 10.1080/00207543.2022.2128922.

[37] A. Kocher, A. Hayward, and A. Fay, “Model-Based Engineering of CPPS Functions and Code Generation for Skills,” in 2022 IEEE 5th International Conference on Industrial Cyber-Physical Systems (ICPS), Coventry, United Kingdom, uuuu-uuuu, pp. 1–8.

[38] M. M. Strljic, O. Riedel, and A. Lechler, “Collective Cloud Manufacturing for Maintaining Diversity in Production through Digital Transformation,” in 2019 IEEE 21st Conference on Business Informatics (CBI), 2019.

[39] M. M. Strljic and O. Riedel, “An evolutionary data model for the implementation of collective cloud manufacturing to maintain individual value-added networks,” in 2019 IEEE 39th Central America and Panama Convention (CONCAPAN XXXIX), 2019.

[40] G. Lanza et al., “Agiles Produktionssystem mittels lernender Roboter bei ungewissen Produktzuständen am Beispiel der Anlasser-Demontage,” at - Automatisierungstechnik, vol. 70, no. 6, pp. 504–516, 2022, doi: 10.1515/auto-2021-0158.

[41] M. Wurster, B. Häfner, D. Gauder, N. Stricker, and G. Lanza, “Fluid Automation — A Definition and an Application in Remanufacturing Production Systems,” Procedia CIRP, vol. 97, pp. 508–513, 2021, doi: 10.1016/j.procir.2020.05.267.

Biography

Sebastian Behrendt (*1997) is a research associate at the Institute of Production Science (wbk) at the Karlsruhe Institute of Technology (KIT) and works in the group of production system planning. He studied mechanical engineering with a specialization in production systems, data analysis and machine learning at the Karlsruhe Institute of Technology (KIT).

Marco Wurster (*1993) is a research associate at the Institute of Production Science (wbk) at the Karlsruhe Institute of Technology (KIT) working in the group of production system planning. In his research he focuses on production planning and control of agile production systems in environments with a high degree of uncertainty such as remanufacturing and product disassembly.

Matthias Milan Strljic (*1991) is a research assistant at the Institute for Control Engineering of Machine Tools and Manufacturing Units, University of Stuttgart in Germany. Strljic received his M.Sc. from the University of Stuttgart. His research interests include cloud manufacturing, production process planning, and production automation.

Jan-Felix Klein (*1992) is a research associate at the Institute for Material Handling and Logistics (IFL) and group lead of the Robotic and Interactive Systems group at the Karlsruhe Institute of Technology (KIT). In his research he focuses on autonomous intralogistics and material handling in highly modular production systems, especially for remanufacturing and automated disassembly systems.

Marvin Carl May (*1994) is a researcher at the Institute for Production Science (wbk) at the Karlsruhe Institute of Technology (KIT). His research concerns production system optimization on strategical, tactical and operational levels by means of intelligent algorithms. The focus lies on discrete manufacturing, in close cooperation with industry, e.g. semiconductor manufacturing, and aims at researching and implementing innovative and application-oriented solutions to technological problems.

Gisela Lanza (*1973) is member of the management board at the Institute of Production Science (wbk) of the Karlsruhe Institute of Technology (KIT). She heads the Production Systems division dealing with the topics of global production strategies, production system planning, and quality assurance in research and industrial practice. Her research focus is on the holistic design and evaluation of production systems.