# Transformability in Material Flow Systems: Towards an Improved Product Development Process

Maximilian Wuennenberg[1], Pia Vollmuth[1], Josef Xu[1], Johannes Fottner[1] and Birgit Vogel-Heuser[2]

[1] Chair of Materials Handling, Material Flow, Logistics
[2] Institute of Automation and Information Systems
TUM School of Engineering and Design
Technical University of Munich
Boltzmannstraße 15, 85748 Garching bei Muenchen, Germany
E-Mail: max.wuennenberg@tum.de

**Abstract.** In a turbulent environment with changing conditions and requirements and the advance of Industry 4.0, transformability is an important aspect for material flow systems along their entire product life cycle. It must be considered already in early development phases and plays a key role in the operation of a system up to eventual retrofits. Therefore, transformability can help in making material flow systems reusable and thus more sustainable. Developments in the state of the art make change management a challenging task since specifications from several mechatronic domains need to be considered in a multi-disciplinary project environment. This paper analyzes established approaches that have been developed in research works, and combines these findings with the view of practitioners and thereby deduces a collection of requirements for the consistent development and operation of material flow systems. These requirements cover necessary models of the system and the participants in the development process, as well as the consideration of life cycle-related aspects for all components. After that, it is discussed which requirements are already partially met by existing approaches and which aspects need to be developed in the future to reach the objective of improved transformability in the product life cycle of material flow systems.

**Keywords:** Consistency Management, Flexibility, Internal Logistics, Material Flow Systems, Optimization, Retrofits, Transformability.

## 1 Introduction

Manufacturing companies face increasing challenges regarding volatile demands and rapidly changing product portfolios. One of the key drivers to deal with these challenges of changing requirements is successful internal logistics. From a technical point of view, this is realized with Material Flow Systems (MFS). MFS are designed to respond flexibly to a certain set of predefined and predicted requirements. In the event of unplannable changes during the life cycle, transformable MFS represent a competitive advantage, as they can adapt proactively and outside of preconceived margins of action [1]. Hence, they form an important part of modern Industry 4.0 environments. Many technical and organizational factors influence the – both analog and digital – transformations of MFS and thus these processes are highly complex and require a set of different competences in planning. However, transformation processes are often carried out manually and very case-specific, which leads to increased effort and isolated solutions. As MFS have to undergo various transformations within their lifecycle, a systematic approach for the consistent planning of the transformation process is of high importance.

The development of MFS requires the participation of specialists from several domains, such as mechanical engineering and software engineering. The resulting multidisciplinary teams usually have tight project schedules and therefore several specialists must work in parallel. At the same time, the development tasks are often divided across enterprises in the sense that each company is responsible for the development of a certain group of modules. The resulting project organization is highly complex and contains many dependencies between the participants [2]. A survey between 25 practitioners revealed that nonetheless, every specialist needs to work in a specific developing environment, which means he is using software tools that are usually only used in his domain [3]. Although each of these tools has its own subset of properties of MFS that it can depict, information overlaps between several tools are possible. However, the modeled information is not automatically equivalent in the entire project. Resulting contradictions are referred to as inconsistencies [4]. Inconsistencies can cause delays in the development process and lead to additional working effort as well as unforeseeable alterations in the project schedule. The reason for this is that development steps must be repeatedly executed to make sure that all models and documents of the MFS are free of contradictions. Otherwise, it can remain obscured that certain modules of the system are incompatible with others until the eventual assembly of the system on the customer's site.

Since MFS have long product life cycles, the need for consistent and transformable models does not cease to exist when the development process is finished. Instead, operators need to realize frequent updates in the system. [5, 6] When the MFS have reached the end of their initial life cycle, there is an alternative to a complete replacement of the system. That is, only some components are replaced to make sure the existing system can still fulfill its current requirements. Such adaptations, which extend the product life cycle, are called retrofits [7]. Successful retrofits offer a significant potential to save investment costs for the manufacturing company and increase the sustainability due to the fact that system components remain reusable. However, they require both a high transformability of the system and its components as well as a documentation which is free of inconsistencies. Although model consistency is a valuable property of MFS, which can help to react when boundary conditions change and to realize retrofits, so far only little attention is paid to this important development paradigm. In many companies that develop MFS, more emphasis lies on other aspects of the development process such as costs and reliability rather than their ability to be integrated into a consistency management framework. That is why, according to a comparative survey among practitioners from the MFS domain, the majority of practitioners report regular coordination problems in their projects [3].

As a conclusion, it can be stated that the Status Quo in the development of MFS is not satisfactory regarding digital and physical transformability in later phases of the life cycle. An actionable procedure to follow when developing MFS is therefore desirable. To ensure this, the first step is the gathering of important requirements for a tool that provides assistance for practitioners. This paper provides such requirements based on the view of practitioners as well as a literature review.

## 2 Research Methodology

This article collects requirements for the development of MFS so that transformability is enabled. As Figure 1 shows, this is achieved by analyzing the state of the art, which consists of contributions to the research literature as well as industry-related studies. The state of the art therefore covers the areas retrofits, transformability, and consistency management, which are linked together by the product life cycle of MFS (colored part of the figure). From that point, a transformability-driven development framework for MFS can be derived and evaluated in further works.

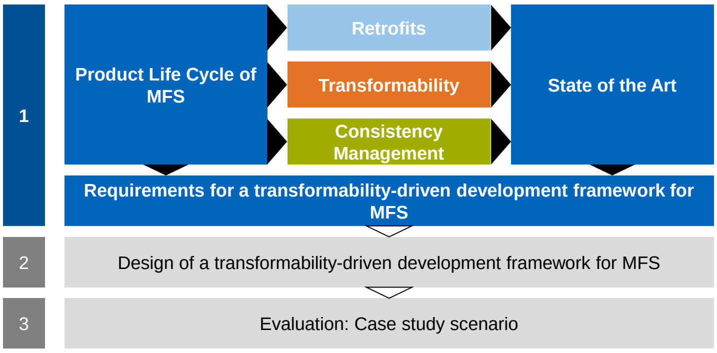


**Fig. 1:** Road map to the development of a transformability-driven product development framework for MFS

The requirements are identified as follows. By analyzing the relevant literature in each domain of the state of the art, key aspects are identified [8]. Those lead to a collection of issues that need to be considered when developing MFS. After that, it is examined what solutions for these issues are most prevalent or can be deduced by merging several solutions. Being oriented towards these solutions, the requirements are formulated.

The development of MFS requires multi-disciplinarity , the developed products have a high degree of digitization, and their life cycle is rather long. These special characteristics of MFS lead to the need for a rather specialized product development framework.

The contribution of this article to the state of the art is a list of requirements for transformability-driven MFS development, which can be used as a checklist for practitioners when revising their current development procedures. That is, by meeting these requirements in the product development process, the later phases of the product life cycle (e.g., retrofits) benefit from this.

## 3 State of the Art

In this article, MFS are considered as the collection of all elements that are responsible for the flow of goods within an enterprise. This covers stationary conveying modules such as roller conveyors or belt conveyors, as well as vehicles such as forklift trucks or tugger trains. Apart from those physical components, the related software is covered by the term MFS as well. This incorporates all digital modules from Programmable Logic Controllers (PLCs) up to the Enterprise Resource Planning (ERP).

Retrofits are an important scenario where the transformability of MFS is challenged. When they take place, they usually come with large efforts because interfaces of the systems often do not fit together with new components (e.g., bus systems, cabling, etc.). Often, this could not be considered during the initial planning of the system. Transformability yields the potential for paying attention to these aspects, thereby enabling structural changes of MFS. Consistency management is the technological enabler to ensure appropriate interfaces. When there are inconsistencies in an assembled, operational system, they often show up not before a major change request is created, for instance due to a retrofit. Thus, enablers for transformability follow similar design principles as enablers for consistency management.

### 3.1 Development and Product Life Cycle of MFS

The product life cycle of MFS contains the following steps (cf. Figure 2, right). It starts with the initial planning phase, which is executed on a high abstraction level. The sales engineers and project planner of the general contractor are in close contact with the customer (i.e., the operator). After defining a set of requirements, preliminary simulation models and solution designs with a growing degree of detail are developed iteratively. In this phase the significant properties of the future system are defined [9]. In the production step all the necessary components for the MFS are shipped to the customer, where the mechanical components and wirings are assembled, and the program code is installed [2]. These first three phases of the life cycle form the product development process. In the next step, product use, the fulfillment of all requirements is extensively tested. Subsequently, the system is operated until it is modernized for extended usage, replaced by a new one or decommissioned [2].

Realizing a development process on different degrees of detail enables more agile project management. Instead of executing all tasks subsequently, several specialists can work on different issues independently and at the same time, which is referred to as Simultaneous Engineering [10]. Even shorter development cycles are possible when the same task is distributed in a team. That is called concurrent engineering [10].

Although both paradigms promise shorter development cycles, they also increase the risk of inconsistencies as many people are working on the same system, but not at the same model. Since there is a need for a consistent documentation of the data and the respective processes to make sure that this potential can be used, data scientists rely on software for database management, as well as higher programming languages for data science and the implementation of artificial intelligence (AI) [11]. Those tools can model different aspects of the system, but there are overlaps as some information exists in more than one model [4]. Whenever this is the case, inconsistencies can occur. The conclusion of these findings for the product development process is as follows: The system architecture needs to be described in a technology-oriented way so that a connection between the desired material flow functionalities and the system behavior can be drawn.

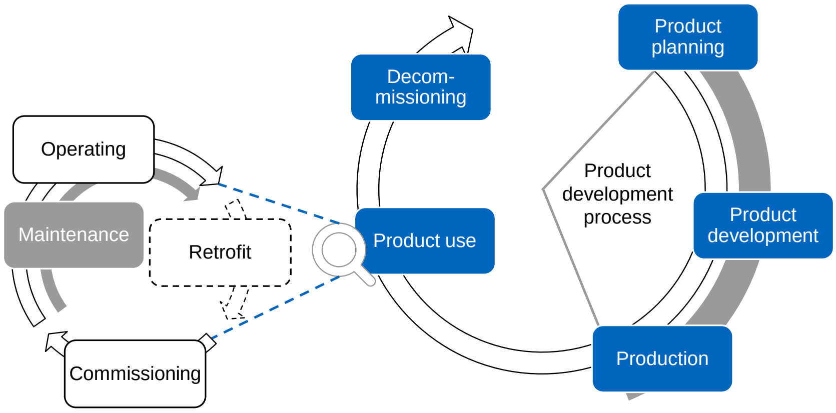


**Fig. 2.** right – Product life cycle, adapted from [2]; left – Product use life cycle of a MFS, based on [2]

To depict the modeling process itself, with the different engineering disciplines involved as well as the components of the eventual system, a proper communication model is necessary [12]. It must be highlighted which stakeholders are working together with whom, how the cooperation is structured in terms of information exchange, and which system models are thereby generated. Model dependency maps are one possible instrument. They work as a base for interdisciplinary communication and collaboration and help to increase the mutual understanding of development tasks [13]. Another approach is the modeling language Business Process Model and Notation (BPMN). With its recent extensions such as BPMN++, cooperation in inter-disciplinary teams can be described, but there is even the possibility to make quantitative comparisons between different process alternatives [4]. As a conclusion, it can be stated that the root causes of system alterations and their respective effects need to be documented during a product development process. This also encompasses the intensities and directions of dependencies.

### 3.2 Retrofits

Intralogistics systems with high level of automation have a lifetime in operation of several decades [14]. To ensure an efficient performance level for as long as possible at low operation costs, maintenance and repair activities are carried out. Nevertheless, modifications to these systems are inevitable due to multiple reasons. New customer requirements or change in legislation can lead to necessary adaptions [15]. When the lifetime of plant components is reached, the failure rate in the intralogistics plant grows. [16]. The replacement of old components can lead to longer downtimes. Instead of building a new MFS, a retrofit (cf. Figure 2, left) can be carried out, in which the components of existing facilities are brought up to a newer technology [7]. The conclusion of this is that components need to be categorized, regarding their role in the life cycle, so that a component-wise planning of the product is made possible.

### 3.3 Transformability

Since requirements for MFS change over their life cycle, and this must be responded to, the systems must be designed transformable. Transformability is a system's ability to adjust reactively or proactively with structural changes to dynamic internal or external factors of influence [17]. The transformation might cover the dimensions of organizational, technological or software alterations. Transformability includes related concepts such as reconfigurability, changeover-ability, flexibility and agility [19]. Reconfigurability focuses on special logistical facilities with autonomous and standardized functional units in order to obtain new machine configurations within a very short time, e.g., Plug-and-Produce modules. changeover-ability describes the ability of a production system to carry out different production processes with the inclusion of set-up work [19]. Within certain predefined dimensions and scopes of action (flexibility corridor), flexible systems can adjust reactively to changing indicators [18]. However, as soon as more extensive, and structural adjustments are necessary, and thus a shift or scaling of the bandwidth or position of the flexibility corridors, transformable systems with no explicit limits and largely solution-neutral preconceived free scopes are required (cf. Figure 3) [18]. The term agility is used in the literature to describe the comprehensive strategic adaptions on a network level e.g., to new markets [19].

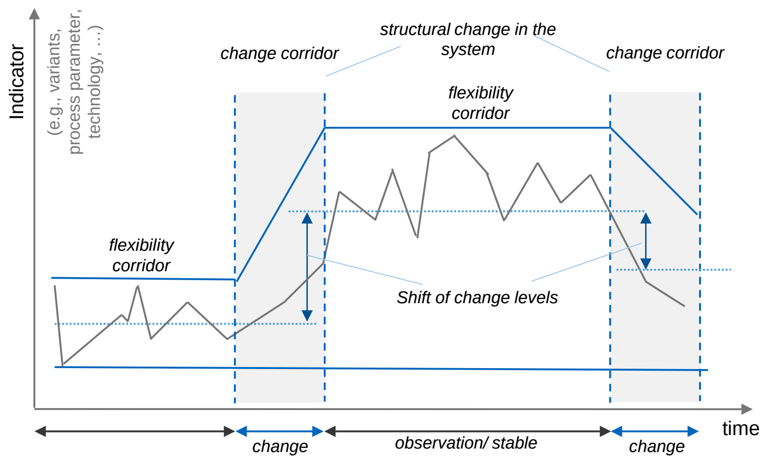


**Fig. 3.** Transformability via shifting and expanding the flexibility corridors, based on [17]

A central role in a system's transformability is taken by change enablers. They describe an inherent property that can be activated in a certain change period and cause the desired change. The change enablers can be summarized in five terms: *Universality* encompasses the dimensioning and design for different requirements [19]. *Modularity* is made feasible through standardized, independent functional units e.g., Plug-and-Produce modules [19]. *Compatibility* is another factor for interconnectivity in terms of material, information, media and energy through uniform software interfaces, qualification structures and compliance with the system requirements [19]. In addition, neutrality contributes to create transformability, as objects do not to influence the properties of other objects [19]. *Mobility* enhances the ability to change the systems components in terms of spatial movability [19]. Related to that, *scalability* encompasses the ability both to add or remove technical, organizational, or spatial resources to extend or limit the performance of a system [19].

The key for transformability of MFS is the modularization of hardware and software as well as the use of module clusters with a central coordinator for information exchange with external modules instead of conventional distributed controls with a high communication effort [20].

As the planning and realization of transformations are often knowledge-dependent, time-consuming, and error-prone, the importance of a systematic approach for the change process has to be highlighted [1]. Especially for complex systems with long lifetimes, the requirements are likely to change over time. To this end, dynamic influencing factors must be considered at an early stage of planning and possible adaptation scenarios must be thought through in advance [15].

### 3.4 Consistency Management

An important step on the way to improved change management is the standardization of model contexts and possible inconsistencies in a project-independent manner. That is, all properties of the system that can be modeled by one particular development tool must be consolidated in order to determine possible overlaps with other models. Every information that is modeled in more than one tool is redundant and therefore causes potential inconsistencies [4].

Traditionally, the control architecture of MFS is hierarchically structured. A typical paradigm for this is the automation pyramid, where the control software is divided into five layers [21]. The top layers contain information systems for ERP systems and Warehouse Management Systems (WMS) [22]. Below that, a Material Flow Computer (MFC) is responsible for the actual routing of Transportation Units (TUs) [22]. The routing commands are sent to the second-lowest layers, which contains PLCs. The PLCs then communicate with sensors and actuators [22]. Those form the lowest layer of the pyramid and directly interact with the physical processes. The key characteristic of this hierarchical architecture is that communication is only happening vertically. Higher layers give orders to lower layers, and lower layers send reports over processes back to the higher layers. Entities on the same layer do not communicate with each other.

However, the need for a different type of control architecture rises with the increasing popularity of agent-based, AI-operated systems [23]. These make MFS more flexible and enable quick responses to alterations in the external conditions. The entire system is divided into partially autonomous modules [23]. Usually, every module of the system is controlled by a designated software agent that is responsible for the flow of TUs in this part of the MFS [24]. Agents communicate with each other on the same level and search for mutual agreements [29]. Optimization methods make sure that collaborative decision-making leads to better outcomes than isolated optimization by each agent. [25] It is proven that decentralized controls enable a better handling of increasing complexity in the manufacturing and logistics environment [25].

Depending on the different types of possible system structures, there are also several ways of how MFS elements can be combined to modules. Integration of distributed components means that modules consist of parts which are located in different regions of the system [26]. Modularized integration groups all components together for each module [26]. Finally, spatial integration means that elements are bundled in a single casing [26]. Summarizing this, it can be stated that during the product development process, a suitable and modularized architecture for the system must be determined.

## 4 Results: Requirements for a transformability-driven product development framework

Summarizing the state of the art shows shortcomings within the established practice of how MFS are developed. Sometimes, there is the goal of a better transformability in the product use phase, for instance with the objective of a retrofit in mind. However, mostly individual aspects of the product life cycle are covered instead of having a multi-lateral development framework. Creating a transformability-oriented method for the product development process of MFS can be achieved as a synthesis of well-established development paradigms, combining their individual advantages with additional elements that are necessary to meet the objective of enhanced transformability. Covering all aspects of the state of the art yields five requirements (R) that need to be fulfilled by a product development framework of MFS, which are as follows (cf. Figure 4):

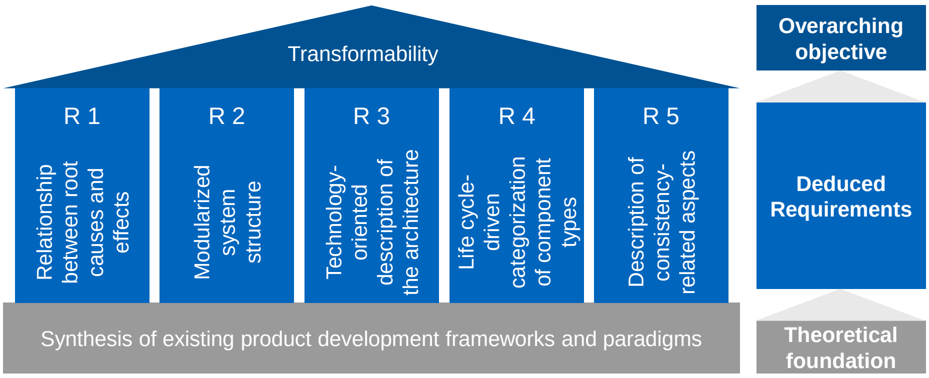


**Fig. 4.** Requirements for a transformability-driven product development process of MFS

**R1: Creating a project view-based relationship between root causes and effects in the product life cycle.** This means a model is required that makes clear which participant influences the development process from the perspective of another specialist, considering the bigger picture of the overall development process. A typical example is the functionality of the system in the shape of process paths, which are created by material flow planners and then serve as major input for the work of software developers. Transparency about the process chain in the development process can only be generated when a model for these dependencies exists.

**R2: Structuring the MFS in a modularized system architecture on multiple layers using a suitable encapsulation of components.** Traditionally, the structure of MFS usually follows a linear, layer-based architecture. With the increasing popularity of decentralized, AI-driven systems, the linear approach is often replaced by a one where within a certain layer, individual elements of the system can and have to interact with each other. To make sure that the architecture and system design supports this interaction as good as possible, the right choice of module encapsulation plays an import role. Instead of distributed modules, where elements that belong together are located in different areas of the system, a high transformability can best be pursued by combining all related components in a single casing. Thereby, an easy exchange of entire modules is made possible. A systematic approach is to assign the elementary material flow operations (conveying, distributing, and merging, handling) to each module [27]. Modularity is considered to be a major enabler for transformability. In addition to that, modularized components can be re-used in various planning scenarios and are therefore system-neutral. Hence, they follow the Plug-and-Produce paradigm where modules should be operational after installation without the necessity for major configurations.

**R3: Providing a technology-oriented description of the architecture in a suitable system model.** A solution is desired where the relationships between all individual, domain-specific models of each module of MFS are described. Thereby, it becomes obvious where interdependencies exist that might lead to contradicting model information. When individual components of the system are changed, certain parameters are influenced, which subsequently impact parameters of other modules in related models. It is important to keep track of these interdependencies to schedule all other adaptions which become necessary when a component is changed. Especially when following development paradigms such as simultaneous engineering or concurrent engineering, keeping the relationships between involved domain-specific models in mind is of high importance. Depicting these relations is an important requirement for a developing framework as it helps to reach a major enabler for transformability: compatibility.

**R4: Systematic and life cycle-driven categorization of component types**, also considering the availability of spare parts**.** The individual components of MFS have different life expectancies. Also, the suppliers of the components offer different timespans in which spare parts are available. This means that changes in the system must be scheduled taking these differences into account. Otherwise, changes or especially retrofits can be impaired because certain components can either not be replaced, or they reach the end of their life cycle significantly earlier than the remaining components. For a certain point in the future, it needs to be clarified in advance which components of the system will have reached the end of their life expectancies, and how well spare parts will be available. If a documentation like this is ensured, transformability-related operations in the future (especially retrofits) can be enhanced.

**R5: Description of consistency-related aspects of the system in a suitable model.** This encompasses every relationship between models and the respective project participants where information overlap can occur. If information is not provided consistently over all models and along the entire product life cycle, the success of system alterations is endangered. Such consistency-related issues cover, for instance, redundant descriptions of elements in certain domain-specific models. To ensure successful changes, in any phase of the MFS's life cycle, information in these models must not contradict each other. This does also concern the connection of different modules of the system, for example in a layout or a process flow chart.

These five requirements form the fundamentals of a method for the product development process of MFS where the enablement of future changes, especially retrofits, is considered. Thus, both the objectives of digital transformation as well as sustainability can be met by fulfilling those requirements.

## 5 Conclusions

In this paper, the state of the art in retrofits, transformable MFS and consistency management was examined and overlaps between these domains were discovered, with the product life cycle of MFS as the linking element. Shortcomings in the common practice for the development of these systems were summarized. From that, five requirements were deduced that need to be fulfilled by a product development method of MFS. The focus was put to enhance the transformability of operational systems, with the eventual goal of optimized retrofits in mind as well. Thereby, the reusability of system components can be increased which is beneficial for the sustainability of the system. Consistency management can work as a key tool to make sure that expectations regarding the transformability can be met. In order to make sure that these findings can lead to advantages for practitioners and thus bring manufacturing companies closer to the goals of Industry 4.0, further research work is necessary (cf. Figure. 1).

To this end, the next important aspect is the introduction of a transformability-driven product development framework that fulfills those requirements and thus also enables the digital transformation of MFS. This method needs to continuously model the relationship between root causes and effects regarding all specialists involved in the product development, to categorize the component types considering their life cycle, and to describe consistency-related aspects. Further on, a modularized system architecture for MFS needs to be selected. These requirements can work as guidelines when selecting individual parts of the framework. Since most elements have already been proposed in the literature, a part of the method can be developed as a synthesis of those. In particular, the following aspects need new approaches: the modeling of dependencies between components and their parameters for when changes are necessary, and a connection to the life cycles of those individual components. Subsequently, the developed approach must be validated by applying it in a real-life change scenario in MFS, for instance in a retrofit project.

# 6 References


1. Marks, P., Hoang, X. L., Weyrich, M., Fay, A.: A systematic approach for supporting the adaptation process of discrete manufacturing machines. Research in Engineering Design 29(4), 621–641 (2018).
2. Verein Deutscher Ingenieure: Design of technical products and systems – a model for product design. VDI No. 2221 Part 1 (2019).
3. Wünnenberg, M., Hujo, D., Schypula, R., Fottner, J., Goedicke, M., Vogel-Heuser, B.: Modellkonsistenz in der Entwicklung von Materialflusssystemen. Zeitschrift für wirtschaftlichen Fabrikbetrieb 116(11), 820–825 (2021).
4. Vogel-Heuser, B., Reif, J. A. M., Passoth, J.-H., Huber, C., Brodbeck, F. C., Maasen, S., Lindemann, U., Hujo, D.: BPMN++ to support managing organisational, multiteam and systems engineering aspects in cyber physical production systems design and operation. Design Science, 8, (2022).
5. Kernschmidt, K., Feldmann, S., Vogel-Heuser, B.: A model-based framework for increasing the interdisciplinary design of mechatronic production systems. Journal of Engineering Design 29(11), 617–643 (2018).
6. Pecht, M. G., Sandborn, P. A., Solomon, R.: Electronic part life cycle concepts and obsolescence forecasting. IEEE Trans. Comp. Packag. Technol. 23(4), 707–717 (2000).
7. Xu, J., Fottner, J.: Retrofits von Regalbediengeräten. ZWF Zeitschrift für wirtschaftlichen Fabrikbetrieb 116(7-8), 501–504 (2021).
8. Seuring, S., Gold, S.: Conducting content-analysis based literature reviews in supply chain management. Supply Chain Management: An International Journal 17 (5), 544–555 (2012).
9. Baker, P., Canessa, M.: Warehouse design: A structured approach. European Journal of Operational Research 193(2), 425–436 (2009).
10. Verein Deutscher Ingenieure / Verband der Elektrotechnik, Elektronik, Informationstechnik: Development of Cyber-Physical Mechatronic Systems. VDI/VDE No. 2206 (2020).
11. Ungermann, F., Kuhnle, A., Stricker, N., Lanza, G.: Data Analytics for Manufacturing Systems – A Data-Driven Approach for Process Optimization. Procedia CIRP 81, 369–374 (2019).
12. Vogel-Heuser, B., Konersmann, M., Aicher, T., Fischer, J., Ocker, F., Goedicke, M.: Supporting evolution of automated Material Flow Systems as part of CPPS by using coupled meta models. In: Proceedings of the 2018 IEEE International Conference on Industrial Cyber-Physical Systems, pp. 316–323. Piscataway, NJ (2018).
13. Friedl, M., Weingartner, L., Hehenberger, P., Scheidl, R.: Model Dependency Maps for transparent concurrent engineering processes. In: Proceedings of the 14th Mechatronics Forum International Conference, Mechatronics 2014 (2014).
14. Regulin, D., Aicher, T., Vogel-Heuser, B.: Improving Transferability Between Different Engineering Stages in the Development of Automated Material Flow Modules. IEEE Transactions on Automation Science and Engineering 13(4), 1422–1432 (2016).
15. Vogel-Heuser, B., Fay, A., Schaefer, I., Tichy, M.: Evolution of software in automated production systems: Challenges and research directions. Journal of Systems and Software 110, 54–84 (2015).
16. Zio, E.: Reliability engineering: Old problems and new challenges. Reliability Engineering & System Safety 94(2), 125–141 (2009).
17. Siddiqi, A., Weck, O. L. de: Modeling Methods and Conceptual Design Principles for Reconfigurable Systems. Journal of Mechanical Design 130(10) (2008).
18. Verein Deutscher Ingenieure: Transformability. VDI No. 5201 (2017).

19. Wiendahl, H., Reichardt, J., Nyhuis, P.: Handbook Factory Planning and Design. Springer Berlin Heidelberg, Berlin (2015).
20. Lieberoth-Leden, C., Regulin, D., Günthner, W. A.: Efficient Messaging through Cluster Coordinators in Decentralized Controlled Material Flow Systems. MATEC Web of Conferences 81, 6005 (2016).
21. Verban Deutscher Maschinen- und Anlagenbau: Data Interfaces in Material Flow Controls. VDMA No. 15276 (1994).
22. Deutsches Institut für Normung: Enterprise-control system integration – Part 1: Models and terminology. DIN No. 62264 Part 1 (2014).
23. Fischer, J., Marcos, M., Vogel-Heuser, B.: Model-based development of a multi-agent system for controlling material flow systems. at – Automatisierungstechnik 66(5), 438–448 (2018).
24. Fischer, J., Lieberoth-Leden, C., Fottner, J., Vogel-Heuser, B.: Design, Application, and Evaluation of a Multiagent System in the Logistics Domain. IEEE Transactions on Automation Science and Engineering 17(3), 1283–1296 (2020).
25. Gehlhoff, F., Fay, A.: On agent-based decentralized and integrated scheduling for small-scale manufacturing. at – Automatisierungstechnik 68(1), 15–31 (2020).
26. Verein Deutscher Ingenieure: Methodology for the development of mechatronic systems. VDI No. 2206 (2004).
27. Spindler, M., Aicher, T., Schütz, D., Vogel-Heuser, B., Günthner, W. A.: Modularized Control Algorithm for Automated Material Handling Systems. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC). Rio de Janeiro (2016).