





Product family engineering along the life cycle – research aspects to cope with variability in advanced systems

Markus Christian Berschik , Marc Zuefle , Fabian Niklas Laukotka  and Dieter Krause 

Institute of Product Development and Mechanical Engineering Design, Hamburg University of Technology, Hamburg, Germany

Abstract

The evolutionary development of advanced systems (AS) leads to a necessary rethinking of how they can be supported methodically and in terms of processes in product development. Advanced systems engineering (ASE) offers a novel and holistically adaptive approach to facing such challenges in a structured way. However, many of the ASE use cases relate to the development of systems as products, product networks or individual projects. The additional consideration of entire modular product families within AS offers a further decisive advantage for companies, organisations and the people in ASE. By considering modular product families along the entire life cycle in a product family engineering (PFE), the approaches of ASE can bring their impact and potential to additional system levels occurring when considering product families. The systems, which become complex through variety and collaboration, are broken down into their system elements in a structured way and prepared for a common interdisciplinary understanding, as conveyed by ASE. In this paper, the PFE is presented in excerpts using examples of various aspects and points in time of the product's life as a complementary approach for ASE.

Keywords: MBSE, Variety, Modeling, Product family, Advanced systems

Received 11 August 2023

Revised 12 June 2024

Accepted 18 June 2024

Corresponding author

M. C. Berschik

markus.berschik@tuhh.de

© The Author(s), 2024. Published by Cambridge University Press. This is an Open Access article, distributed under the terms of the Creative Commons Attribution licence (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted re-use, distribution and reproduction, provided the original article is properly cited.

Des. Sci., vol. 10, e27

journals.cambridge.org/dsj

DOI: [10.1017/dsj.2024.21](https://doi.org/10.1017/dsj.2024.21)

the **Design Society**
a worldwide community

 **CAMBRIDGE**
UNIVERSITY PRESS

1. Introduction and motivation

Rapid advances in technology and the ever-growing networking and digitalisation of systems are megatrends that have ushered in a new era of product development. Megatrends continue to play a decisive role in product development. Various examples, such as globalisation, servitisation or individualisation, have a long-term influence on business models and their implementation in product development (Krause & Gebhardt 2023). The resulting new demands on products create new challenges for their development. Today, simple mechanical products are being replaced by highly complex advanced systems (AS) whose multi-layered interactions and networked functions beyond hardware and software push the limits of previously applied methods and processes. The shift from mechanical-oriented mechatronic products or systems to complex cybernetic systems, including service products, increases the complexity of the various structures, architectures and dependencies within those systems. This leads to a more extensive impact of variety between hardware, software and service structures within the regarded

system context (Blecker & Abdelkafi 2006; ElMaraghy *et al.* 2012; Tomiyama *et al.* 2019). This increase in dependencies requires increasing interdisciplinary cooperation and collaboration within the organisation and corresponding development procedures that allow data exchange across domains (Isermann 2008; Hehenberger *et al.* 2016). While such AS offers extensive added value for the customer and thus for the market, product, process and humans constantly interact with the complexity-increasing scopes (Colfer & Baldwin 2016). Developers and engineers are also challenged to implement the external requirements with the best possible internal variance, which such variant-induced complexity entails (Tukker 2015; Krause & Gebhardt 2023).

In this context, advanced systems engineering (ASE) is required, and the central issue is mastering the growing variety-induced and collaboration-induced complexity in the development of AS (Morgan, Holzer, & Eveleigh 2021; Yassine 2021; Krause & Gebhardt 2023). As an approach to cope with this complexity in AS, modularisation or the development of modular product families is integrated into the context of ASE as an established and effective strategy to manage these challenges.

The derived product family engineering (PFE) allows us to systematically address and control the internal variety and variability within AS caused by the introduced variety-induced complexity. Different system derivatives in a product family can be configured from a limited number of system elements through the targeted design of a variant-oriented and modularised system architecture. This enables more efficient development of AS and integrates the entire system life cycle in the internal architecture design and internal collaboration. This differs from the concept of product line engineering (PLE), which is discussed in Section 2.3.

2. Research background

To address and classify the approach of PFE in the context of the respective key areas, the essential contents of ASE and variability in product families from the author's perspective are defined. Subsequently, the fundamentals forming the basis for PFE are presented.

2.1. Systems engineering

The International Council on Systems Engineering (INCOSE) clearly describes systems engineering (SE) as an interdisciplinary approach that “means to enable the realisation of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem [...]” (Walden *et al.* 2015). With rising applications of SE worldwide and in different domains, an ambiguous understanding of SE has developed that has concurrently led to slightly different interpretations and practices of SE (Inkermann 2021). Likewise, from the start, there was a need to support engineers and enable effective coordination of projects (Martin 1997). Clearly described processes that target main engineering activities, such as system requirement definition, architecture definition or verification, are defined by INCOSE (2007) and ISO/EC/IEEE (2015), as well as the widely known V-Modell (VDI/VDE 2021). Additionally, complementary methods define, among other

things, how to decompose tasks or sequence actions as well as the required input information. At that SE is not simply the engineering of systems but a clearly defined set of tasks, processes, and methods.

2.2. Model-based systems engineering

SE and its tasks, processes and methods all come with the associated data that must be documented. With rising project complexity, handling the likewise rising amount of information poses a challenge. This is faced by turning to a more formalised way of storing data, such as semi-formalised modelling languages. This shift to a model-based approach also allows for integrating different views of a system into a single consistent system model.

INCOSE describes this approach as model-based systems engineering (MBSE) and “the formalised application of modelling to support system requirements, design, analysis, verification, and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases. MBSE is part of a long-term trend toward model-centric approaches adopted by other engineering disciplines, including mechanical, electrical and software” (INCOSE 2007). In their INCOSE Handbook, MBSE is listed as one of the cross-cutting SE methods, clearly linking MBSE to SE (Walden *et al.* 2015). Before that and independently from SE as defined by INCOSE, Wymore uses the term proposing system theoretic models and physics-based engineering models to represent organisational problems and “are developed, manipulated and managed by system engineers throughout the lifecycle” (Wymore 1993). The linkage between MBSE and SE is also highlighted by Friedenthal, Moore, & Steiner (2015), who says, “MBSE emphasises the use of models to perform SE activities that are traditionally performed using documents”.

In consequence, MBSE shall not be understood as applying modelling to any (engineering) process but as explicit support of SE activities as defined by INCOSE (2007) and ISO/EC/IEEE (2015), as well as using semiformal system modelling techniques. However, an exploratory study of publications has shown a diverse understanding of SE and MBSE, and the term MBSE is often not used accordingly (Berschik *et al.* 2023). Usually, only some elements related to MBSE are used and applied to different tasks or challenges. In addition to the required linkage to SE and based on literature, (Berschik *et al.* 2023) define four aspects essential for MBSE: the *system model* and the three modelling pillars *modelling method*, *modelling language* and *modelling tool* as defined by Delligatti (2014). The system model can be seen as the connecting element of the described pillars. It integrates different views of the system into one holistic model.

2.3. Initiation of ASE

The so-called ASE is an approach to the development of complex systems. It is defined as integrating technology, people and processes to develop innovative solutions to multi-disciplinary challenges. ASE emerged in response to the growing complexity of technologies and applications in various industries such as aerospace, automotive and information technology (Wegmann, Johnson, & Anderson 2020). This complexity required a systematic approach to understand component interactions better and improve system performance (Liu *et al.* 2017). Integrating

ASE into methodical product development can better analyse, design and optimise complex systems. By systematically considering interactions and interdependencies between components, efficiency improvements and innovations are enabled, leading to advanced products and solutions.

Therefore, ASE is meant to be a holistic approach that integrates the already mentioned SE and considers AS development and advanced engineering (AE) to cope with these novel and uprising challenges. AS are complex, sophisticated systems that have emerged through advances in science and technology. These systems are characterised by their ability to adapt to changing environments, provide high interactivity, and offer innovative solutions to complicated problems. AS have evolved in many areas, including autonomous vehicles, smart healthcare and Industry 4.0, with advances in microelectronics, sensors, data analytics and artificial intelligence (He, Chen, & Lee 2018). The development of AS requires close collaboration between engineers, scientists and other professionals from different disciplines. Methodical product development enables the structured integration of knowledge and expertise to design complex systems (Kim, Park, & Chang 2021). Those challenges in collaboration induce AE. This enhanced way of engineering refers to the application of cutting-edge technologies, methods and materials in the creation of innovative products, systems or processes. AE involves utilising techniques and approaches that surpass traditional engineering practices. The integration of simulation technologies, additive manufacturing, virtual reality and other advancements has paved the way for the development of AE as we know it today (Brown & Smith 2019). With AE, engineers can design intricate and lifelike models of products and systems before physically producing them. These benefits are faster and more cost-effective product development because potential defects can be identified and resolved early (Arora & Raja 2016).

As a more in-depth addition, collaboration in AE is one of the key points in the increased interaction in the development process (Mcharek *et al.* 2019). In AE, the focus is on transferring new methods and approaches to engineering (Mertens *et al.* 2022). As can be seen in Figure 1, the interaction between an exemplary selected set of four disciplines is changing from cooperation to collaboration. On the left side of Figure 1, different participants in the engineering of systems are working parallel and in cooperation on their designs. On the right side, various participants are working, in potential iterative interaction, on a collaborative system design (Schmidt, Weiss, & Paetzold 2018). Those sets of participants of different domains

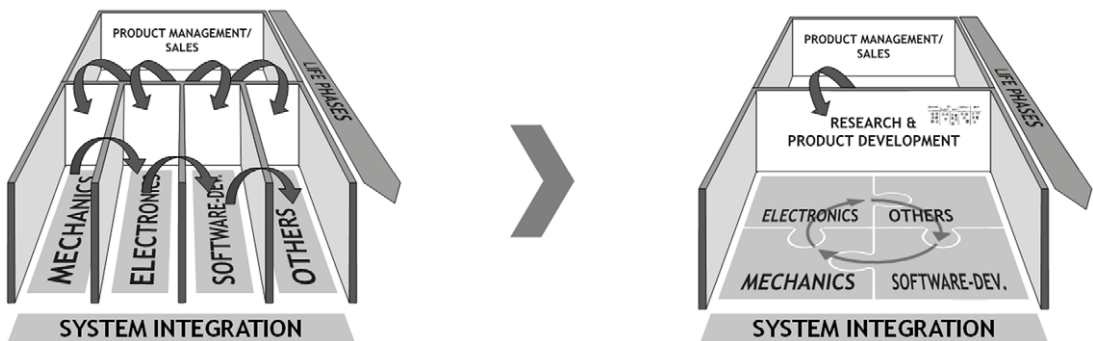


Figure 1. Shift from conventional sequential engineering to collaborative advanced engineering (AE).

and disciplines will vary from specific system to system. A holistic integration of all relevant domains and disciplines is necessary for overall collaboration consideration. [Figure 1](#) shows only a preselection of possible disciplines to present them more clearly.

Due to the goal of PFE as a cross-section methodology, the depicted focus does not include all domains and disciplines integrated into a holistic ASE approach. However, it addresses all relevant core steps for handling internal variety due to collaboration-induced complexity. Collaboration of different domains and development disciplines derives as one essential part of AE. Successful collaboration contributes significantly to the fulfilment of customer requirements and is, therefore, to be regarded as a critical factor for the company's success (Mertens *et al.* 2022). Central prerequisites for this are clear communication between all participants and defined goals and requirements of the product families. Relevant information and resources must be available throughout the entire development process to ensure efficient collaboration and targeted SE (Yassine 2021). Furthermore, a common understanding of the product family's requirements and goals is necessary. Due to continuously increasing requirements and shortened development cycles, as well as the associated complexity of the products, the number of teams and people involved is increasing (Mertens *et al.* 2022).

2.4. Product family and variety

In the literature, the term PLE is often used in the field of software development. It is an overarching term for developing and managing similar software systems under consideration of software variance (Metzger & Pohl 2014). Based on this description, a stronger link to SE was established with the definition of feature-based PLE (ISO/EC/IEEE 2021). Here, different features are used to build up the market view of products as the basis for the resulting configuration of the product set under consideration. The external variety of the systems under consideration is thus described, and the resulting product instances are formed. The analysis of the internal variety within the various product instances is usually not described further. A product is built on predefined domain assets and configured towards the final product asset instance according to the selected features. Hence, already predefined solution patterns can establish different variants (ISO/EC/IEEE 2021). This solution pattern can be used for knowledge management and can therefore provide a basis for the description of variants (Weber & Husung 2016; Anacker *et al.* 2020). The term PLE is sometimes used as a synonym for the term PFE (Hummell & Hause 2015). Although the terms are occasionally synonymous, they can also mean different things. Also, compare [Figure 4](#) in the later section.

Product lines, as considered in PLE, are clustered products within a product program with similar application areas, functionalities, or production processes. They are grouped in a way that makes sense from a business and strategic point of view. They represent a structured grouping of product families that are often defined on a company-specific basis. Product lines help keep track of product development and management (Rupp 1988; Krause & Gebhardt 2023).

A product family refers to a set of product variants with similar functional principles, technologies and the same areas of application or production processes. The product family comprises all variants that differ in at least one property or element but have the same basic functionality (Rupp 1988; Meyer & Lehnerd 1997;

Krause & Gebhardt 2023). Ideally, these differences are relevant for customer decisions, as they enable the selection of the appropriate product variants (Mortensen 1999; Harlou 2006).

Another important feature of product families is that they include all the variants of an offered product. This enables synergies between the components used. Although product families are not necessarily modular, they can be designed modularly to increase efficiency in development and production. This approach is similar to platform design, used mainly in the US (Robertson & Ulrich 1998; Simpson, Siddique, & Jiao 2006; Pirmoradi & Wang 2011). It is not only about product-side implementation but also about integrating action systems and organisational resources for PFE, as shown in Figure 2.

Product families represent a cluster of similar systems that can be developed together. Despite specific distinguishing features, they can exploit synergies and enable a holistic approach to product development. This allows resources to be used more efficiently and development times to be shortened, ultimately leading to more competitive products. Parts of the individual systems can thus be reused in the best possible way in similar systems (Walden *et al.* 2015).

The consideration of variants has diverse aspects, as each variance in a component induces complexity in the system and the associated processes. This has many effects, such as new work steps, special processes, or a decreasing batch size (Krause & Gebhardt 2023). Thus, variant management is a crucial cross-cutting topic in interdisciplinary PFE. In this, the task is to reduce and avoid internal variance while keeping the external variance of the product family at a maximum.

In the field of modelling variant management, there are already various approaches to depicting variance. In software development, for example, the orthogonal variability model (OVM) exists as a concept for mapping variant information (Pohl, Böckle, & Linden 2005). The OVM is used in software development within diagrams of the Unified Modeling Language (UML) and thus offers a visualisation for differentiating program properties and the associated variation

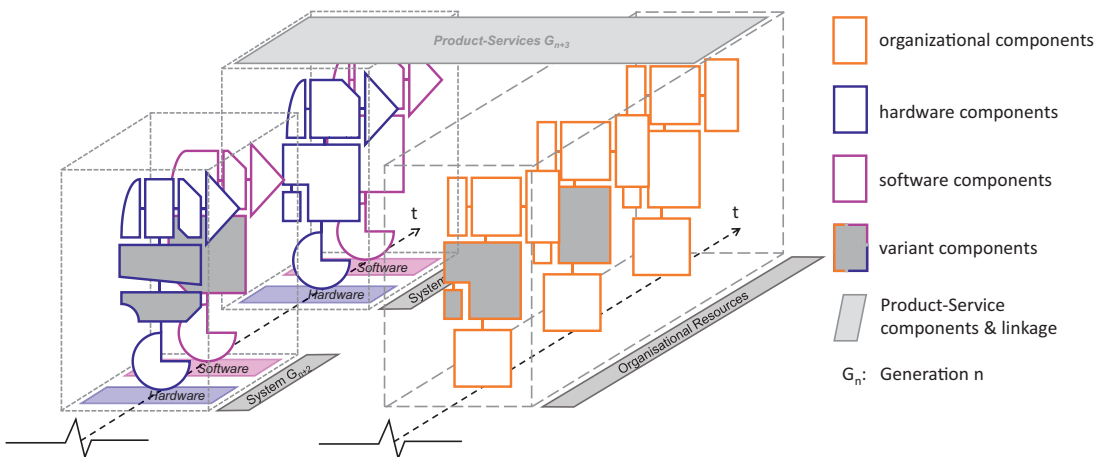


Figure 2. Product family generations with system elements, including hardware and software layers (left). Evolution of linked relevant organisational resources for product family engineering (PFE) (right). Extended by additional Services in the Product Family (top).

(Metzger & Pohl 2014). Furthermore, the common variability language (CVL) of the object management group (Haugen, Wąsowski, & Czarnecki 2012) exists as a cross-disciplinary modelling language for variants. In SE, especially MBSE, there are various profiles or modelling methods for the modelling language SysML, such as the variant modelling with SysML (VAMOS) method. VAMOS supports the modeller in modelling different variants using packages and stereotypes in SysML. The expression of the variance is modelled here on the physical level, i.e., in the individual components and variants (Weilkiens 2016). Other ways to model the variance are with the help of plugins for the modelling tools. Plugins such as model-based product line engineering (MBPLE) for cameo systems modeler can be used to model the behavioural modelling of variants and their associated configuration (Colletti *et al.* 2020).

Product families are suitable for designing modular architectures due to their characteristics in variant creation and management. In this context, modularisation in product families is determined by the properties and characteristics of modularisation, through which product families can adopt different strategic approaches in exploiting different strategies (Mortensen 1999; Harlou 2006).

Modularity is a gradual property of the product structure. What is essential is the concentration of the couplings of the components within the modules. These have stronger couplings to each other than to other components or modules. Modules combine components with certain common features, which can thus be treated as a logical unit.

The modularity is specifically designed to meet the requirements of all product life phases. Particularly noteworthy are

- Reusability: Modules are used in different products to enable savings through economies of scale.
- Combinability: Different product variants can be configured by combining modules.

An essential measure for realisation is interface standardisation and function binding. In the latter, modules fulfil exactly one function or a defined functional scope (Salvador 2007; Krause & Gebhardt 2023). Like modularity, its properties are also considered gradual. In their entirety and considering their interactions, they describe the modularity of a product structure. The goal in the development of modular product families is not necessarily a product structure that is as modular as possible. Instead, the modular properties should be designed to achieve the highest possible company- and product-specific benefit in all life phases of the product family.

3. Research gap and research question of this contribution

ASE has undoubtedly enabled significant advances in the development of products and projects. Research in this field has focused intensively on the holistic consideration of technology, processes and human factors to achieve optimal results. However, one crucial aspect still has potential for further research: the integration of product strategic product families, their induced variability and the resulting effects in different phases of a product's life.

A methodically developed product family comprises a group of related products that share common components but also have different characteristics to suit different customer groups. At this, the strategic consideration of variety throughout the product family is key to managing and reducing costs resulting from product complexity by sharing components, processes and technologies across multiple products. Reusing elements that have already been developed and tested would save time and money and lead to an accelerated time-to-market for new products. By sharing knowledge, experience and best practices between the individual products in a family, companies could continuously optimise their products even in later phases.

Nevertheless, although the concept of integrating product families in ASE seems promising, it also poses challenges: Variety in product families is challenging at different points in the lifetime of a product. During the architecture design, for example, the internal and external variety of the product family needs to be managed, which poses a challenge when considering increasingly large and intricate products or systems. Tests of each product variant are an essential part of the system integration. By managing the variance, a reduction in necessary tests can be achieved. Hence, there is an overarching dependency between the different aspects of development. The variance in product families can be seen as one primary driver for complexity.

The collaboration between different engineering domains is increasing through the increasing multi-disciplinary development. Therefore, collaborative-induced complexity is rising, which also needs to be considered. Another critical aspect is the growing amount of data produced during the development of products. Thus, consistent and holistic availability of strategically relevant data is essential to cope with upcoming challenges, especially when considering AS. Current implementations solely consider the development or focus on aspects from a different hierarchical level, e.g., product lines. Product lines usually focus on similar application areas, functionalities or product processes across multiple product families—often from a business or strategic point of view. On the other hand, numerous approaches focus only on single products—not considering variety sufficiently.

Consequently, there is a research gap in this area of product families in AS. This contribution focuses on the research question:

“How can the explicit and strategic consideration of product families and their variability in the context of ASE improve the development and its further product life cycle?”

For this purpose, product development and usage aspects will be analysed and exemplified. The next chapter will take a closer look at the challenges and potentials of this idea to present possible ways for a successful implementation of product families in ASE.

4. Product families in the context of ASE

Methodically developed product families are a typical way to enable product creators to allow their customers to choose the best matching from a range of different variants (high external variety) while also keeping the complexity of the product architecture manageable for them (low internal variety). This becomes especially important when considering AS, as described in [Section 2.3](#). The explicit inclusion of aspects of life phases after the initial creation simultaneously opens

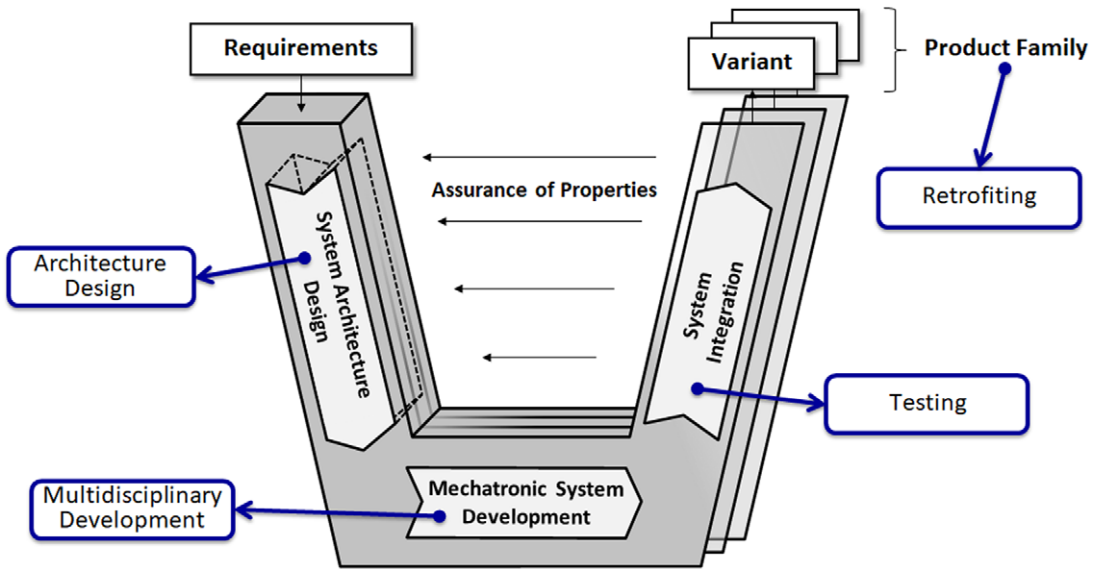


Figure 3. Simplified V-Modell for different product variants resulting in a product family and distinct consideration aspects in this contribution.

new opportunities and creates new challenges. PFE is dedicated to the product-strategic and overarching consideration of different aspects of variability throughout a product’s or system’s life with the holistic management of family-related information. It can be related to the V-Model during the earlier phases, albeit with the additional consideration of the variety of product families.

Additionally, processes of later phases, explicitly during product usage, are holistically integrated into the approach. Hence, it can be seen as a cross-cutting activity. Considered aspects include the architecture design suitable for product families to comprehensive domain-specific models and system testing and maintenance or retrofit tasks during the usage (Figure 3).

This work presents a selection of different aspects of the overarching approach of PFE. Some aspects will be described in more detail, while others will be briefly characterised. It is not to be understood as an all-encompassing selection but as a representative introduction to the approach. They are based on similar ideas and results of past, recent and ongoing research in their respective fields and are now conflated into one continuous approach.

The following sections use an explanatory example of a cyber-physical manual bending product family. This product family includes all similar machines with thin sheet bending as their primary function and shows synergies in technology and functions, as depicted in Figure 4 as the fourth level highlighted in light blue. Within the scope of the higher level production program (Figure 4 second level) and product line (Figure 4 third level), this product family would be expanded to include other bending machines with different functions and technologies, e.g., swivel-arm-robot bending, which would be an additional product family on fourth level in Figure 4. The exemplary product family of manual bending machines focuses on die bending. Here, the sheet metal is deformed by a controlled downward movement of the press beam. A sheet metal is pressed into a matrix by a tool

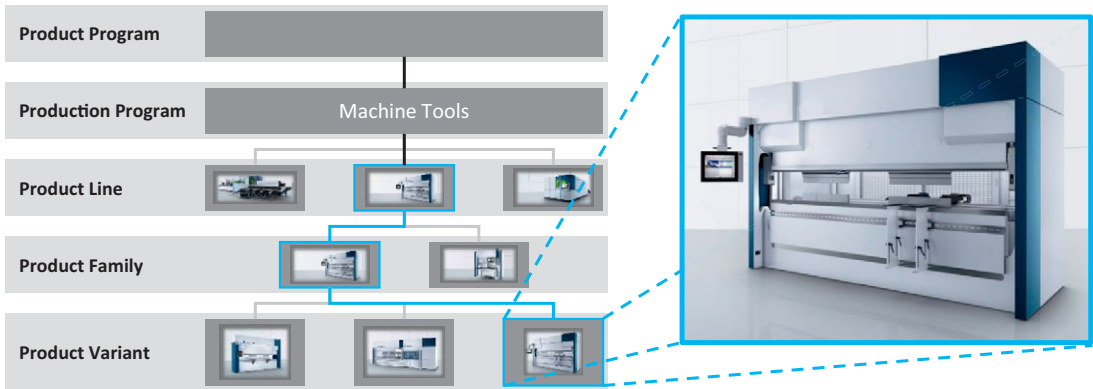


Figure 4. From Product Program to Product Variant using the explanatory example of a TRUMPF Bending Machine. Adapted from (Krause & Gebhardt 2023; TRUMPF).

and formed in a straight line. The installed back gauges ensure an exact position of the workpiece, whereby these can have several axes for supported positioning depending on the design. The product variants result from automation, performance, width and connectivity as variant product properties. Figure 4 shows the representative bending machine variant of the product family on the fifth level compared to other variants and the different levels of hierarchy. The data used here is heavily modified due to its sensitivity.

4.1. Model-based architecture design of product families

In the following section, the architectural design of product families is looked at from a model-based perspective. Hence, it is focused on the modelling of the architecture and the resulting benefits for further usage of the underlying system model.

The so-called V-Model is often used in the development of mechatronic systems. The technical processes of ISO 15288 are also known as SE (ISO/EC/IEEE 2015). They can be classified in the V-model and form the basis for further description. The technical development process records and defines the stakeholders' requirements. Based on this, the requirements are analysed. Both process steps of ISO 15288 are considered in Requirements Engineering and are not described further here (Pohl & Rupp 2011). Many methods exist for Requirements Engineering that have already been implemented model-based, see Holt, Perry, & Brownsword (2012) and Japs (2020). Building on the Requirements Engineering, the architecture design follows. A wide variety of models and structures are used in architectural design. One of them is the system architecture. It is used to link the functional structure of the system with the component structure (Ulrich 1995). In this context, the so-called RFLP-Framework is used to model a system. This approach is used to link the various development data and is based on the interrelationships of the V-Model (Eigner & Gilz 2012; Kleiner & Kramer 2013).

The related development data can be systematically divided and assigned based on the commonly used RFLP-Framework. Building on the left side of the V-Model, all basic development data can be mapped (Eigner & Gilz 2012). When describing the architecture, the variety regarding the system context of the individual data

elements is often not considered. However, it represents a significant cost driver in developing and maintaining product families. For this reason, it should already be considered in the architecture design (Krause & Gebhardt 2023). For further classification, product families can be seen as subsets of a product line; these describe the internal variety of the products under consideration, whereas the product line focuses more on the market view and the resulting external variety (ISO/EC/IEEE 2021).

Based on the described relationships, a higher-level data structure can be established. This can be mapped in an ontology using the modelling language systems modelling language (SysML). It is used to express the entities of different data elements and their relationships on an understandable and abstract level (Aßmann, Zschaler, & Wagner 2006). Figure 5 shows a simplified architecture ontology for product families that depicts the data relationships and their classification in the RFLP framework. This maps the classic functional development data (VDI/VDE 2019) and supplements it with product strategy aspects (shown in light orange colour), for example, by dividing a system into modules and variants. Hence, the architecture focuses on variety and the resulting complexity for the overall product family. Thus, the internal variety on the different levels, such as functional or logical, is modelled and focused in detail.

In contrast to the conventional usage of the RFLP-Framework, PFE uses this framework more extensively to model the allocations between the solution principles and system elements of different domains and disciplines in complex cybernetic systems and services. On a logical level, for example, the RFLP-Framework is extended by the different solution principles of hardware, software and service domains, e.g., electro-technical sequences or sequences and various programmable logic controllers. In addition, the physical level is extended by all

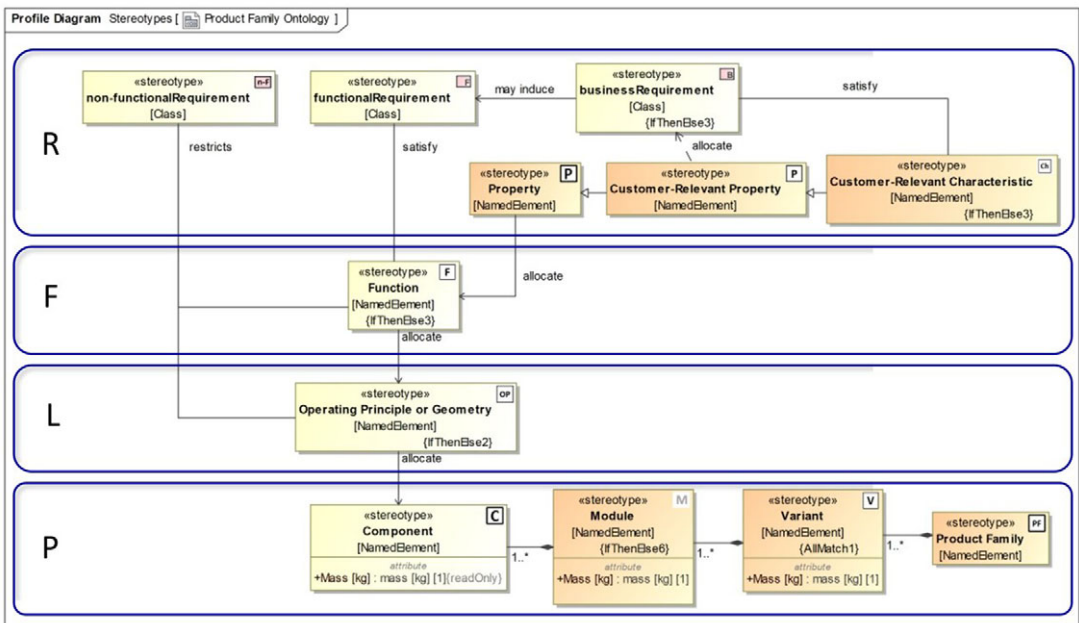


Figure 5. Simplified Ontology of the data model for modelling product families.

elements regarded by the included domains. So, not only hardware, software and service elements can be depicted and addressed, but modules and submodules can also be modelled, integrating all domains.

Product properties are mapped based on requirements starting from the data contexts of the classical development procedures. Product properties describe the characteristics of a product and represent the customer’s perception. They are often not quantifiable and cannot be directly influenced by the developer. Compared to the customer-relevant properties relevant to a customer’s perception of the product, some properties are irrelevant to the customer’s perception. The characteristics of customer-relevant properties can be used as a basis for configuring product variants. Accordingly, only customer-relevant properties should influence the internal variance of the product and hence have a link to business requirements.

Furthermore, internal properties are predetermined by the companies and should not create a variance in the system architecture (Krause & Gebhardt 2023). Several components are combined into product strategic modules or module variants. These, in turn, form the different product variants aggregated into a product family. The modelling implementation was done using the modelling tool Cameo Systems Modeler and based on the modelling language SysML.

In the following, some excerpts of areas are discussed in which a model-based implementation of the architecture of a sheet metal bending machine could benefit the product family’s development and maintenance processes.

Figure 6 shows the sheet metal bending machine product family variants considered in this example. The hierarchical modelling allows the associated modules to be mapped as part properties based on the modelled variants of the product family and the existing components with their variance characteristics to be attached to them.

The system under consideration has four variants, each of which is differentiated by the customer-relevant characteristic of the pressing force. The depth of the sheets to be bent is differentiated into long and short processing areas. In addition, the variants have an optional module that includes a monitoring system for the bending process with additional sensors. The modelled generic modules are thus divided into different module variants with different components attached.

Through the resulting continuous modelling with the help of generalisations and compositions, it is possible to draw holistic conclusions about the various components of the product family. As an example, a selection of components of the modules is shown in Figure 7. The components are shown in the table created based on the model sorted according to their variance and the associated customer-relevant properties.

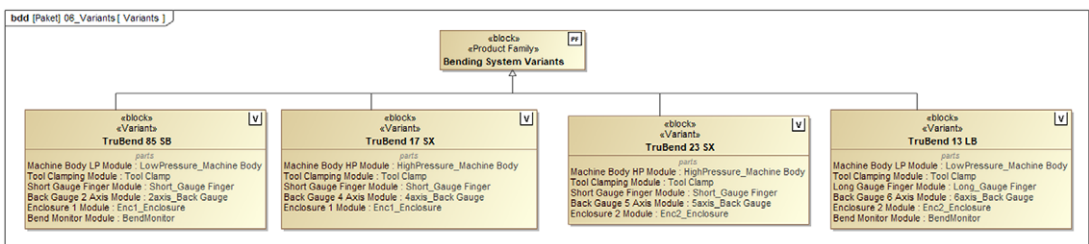


Figure 6. Modelled modules of the exemplary bending machine product family.

#	△ Name	Included Modules	Included Standard Components	Included Variant Components	Included Optional Components	Customer Related Properties
1	TruBend 13 LB	<ul style="list-style-type: none"> M LowPressure_Machine Body M Tool Clamp M Long_Gauge Finger M 6axis_Back Gauge M Enc2_Enclosure M BendMonitor 	<ul style="list-style-type: none"> C Extrusion Punch C Tool Clamp C back housing C roof 	<ul style="list-style-type: none"> C base frame normal C side housing C Back Gauge 6Axis C Base Frame Back Gauge 6Axis 	<ul style="list-style-type: none"> C Bend Monitor Cables C Bend Monitor Screen C Gauge Finger Long Sensor 	<ul style="list-style-type: none"> P Bending Force P Locating Surfaces P Bend Monitor P Gauge Finger Extension
2	TruBend 17 SX	<ul style="list-style-type: none"> M HighPressure_Machine Body M Tool Clamp M Short_Gauge Finger M 4axis_Back Gauge M Enc1_Enclosure 	<ul style="list-style-type: none"> C Extrusion Punch C Tool Clamp C back housing C roof 	<ul style="list-style-type: none"> C base frame heavy C side housing reinforcement C Back Gauge 4Axis C Base Frame Back Gauge 4Axis 		<ul style="list-style-type: none"> P Bending Force P Locating Surfaces
3	TruBend 23 SX	<ul style="list-style-type: none"> M HighPressure_Machine Body M Tool Clamp M Short_Gauge Finger M 5axis_Back Gauge M Enc2_Enclosure 	<ul style="list-style-type: none"> C Extrusion Punch C Tool Clamp C back housing C roof 	<ul style="list-style-type: none"> C base frame heavy C side housing reinforcement C Back Gauge 5Axis C Base Frame Back Gauge 5Axis 		<ul style="list-style-type: none"> P Bending Force P Locating Surfaces
4	TruBend 85 SB	<ul style="list-style-type: none"> M LowPressure_Machine Body M Tool Clamp M Short_Gauge Finger M 2axis_Back Gauge M Enc1_Enclosure M BendMonitor 	<ul style="list-style-type: none"> C Extrusion Punch C Tool Clamp C back housing C roof 	<ul style="list-style-type: none"> C base frame normal C side housing C Base Frame Back Gauge 2Axis C Back Gauge 2Axis 	<ul style="list-style-type: none"> C Bend Monitor Cables C Bend Monitor Screen C Gauge Finger Long Sensor 	<ul style="list-style-type: none"> P Bending Force P Locating Surfaces P Bend Monitor P Gauge Finger Extension

Figure 7. Tabular representation of the modelled components and modules within the individual variants of the bending machine product family.

For example, in the view of the model shown in Figure 7, it is possible to see precisely which components appear in the *TruBend 13 LB* variant. It is possible to see which components appear in the same design across the entire product family and are thus marked as standard components. At the same time, all variants and optional components of the individual variants are displayed. The distinction can also be seen at first glance via the attached icon. The nomenclature that is also used in other tools of the *Integrated PKT-Approach* is used for this (Krause & Gebhardt 2023). Furthermore, the data can be exported as Excel or the standardised CSV format for other programs and visualisation tools.

The dependencies on other development data, such as the functions, can also be drawn through holistic linking. In this way, the connections of functions, via operating principles to components, from a system architecture can be referenced over several levels with the help of metachains, even though no direct connection between functions and components has been modelled. By composing the overall system, the different functions can thus also be mapped to variants and be visualised overall. Dependencies and resulting relationships can be documented more easily in this way, as they can be presented more clearly in different diagrams. For example, an engineer wants to determine which function relates to a particular module in the system architecture because of ongoing customer complaints. With the help of the system model, it is easy to find the module's functions and their derivatives quickly. Due to the consistent model, it is also possible to see in which other module this function is relevant. Consequently, the analysis process can be optimised.

If further connections of the metamodel (Figure 5) are considered, an association of the customer-relevant properties to the variants can also be considered with the overall modelling. This data linking from a sales perspective is a decisive view for a possible configuration in variant management. Here, because of the linkage of the development data of the product family, the indirect reference from properties to variants is traceable without explicitly creating a direct relation. In contrast to classic PLE approaches, the internal variety of the focussed variants and,

thus, the establishment of different reused modules is also of interest and is, therefore, modelled. Building on this representation, a feature-based offer catalogue can be established. Figure 8 shows an extract of customer-relevant properties of the bending machines with their characteristics. These can be selected based on a table like a configurator interface. Plausibility rules are stored for all properties depending on the stored stereotype. After a selection, it can be checked whether all properties required for selecting a variant are set. If a selection is missing, it is automatically displayed in yellow. If an error has occurred, for example, due to multiple selections, this is shown in red.

The variable CRC Input is set in the customer-relevant property by selecting or deselecting the checkbox. This variable is used in the next step to hide unused properties for variant determination.

The linking of the customer-relevant properties with the variants is thus displayed based on the selection and the corresponding hiding of the properties. Like the functions, the customer-relevant properties are also linked to the variants via a meta-chain. Accordingly, changes in the development data directly affect the corresponding link to the variants. Thus, the selection of variants can be implemented directly from the system model without creating another selection model. Figure 9 shows a possible selection of customer-relevant properties. The variant *TrueBend*

#	Customer-relevant Property	Name	CRC Input	Plausibility Check Note
1	P Locating Surfaces	cl _h 2 Axis	<input type="checkbox"/> false	Valid selection.
2	P Locating Surfaces	cl _h 4 Axis	<input type="checkbox"/> false	Valid selection.
3	P Locating Surfaces	cl _h 5 Axis	<input checked="" type="checkbox"/> true	Conflict: Multiple characteristics of the same property selected.
4	P Locating Surfaces	cl _h 6 Axis	<input checked="" type="checkbox"/> true	Conflict: Multiple characteristics of the same property selected.
5	P Bending Force	cl _h 8.5 Tons Bending Pressure	<input type="checkbox"/> false	No selection within the associated property.
6	P Bending Force	cl _h 13.0 Tons Bending Pressure	<input type="checkbox"/> false	No selection within the associated property.
7	P Bending Force	cl _h 17.0 Tons Bending Pressure	<input type="checkbox"/> false	No selection within the associated property.
8	P Bending Force	cl _h 23.0 Tons Bending Pressure	<input type="checkbox"/> false	No selection within the associated property.
9	P Bend Monitor	cl _h Bend Monitor Included	<input checked="" type="checkbox"/> true	Valid selection.
10	P Gauge Finger Extension	ch _h Gauge Finger Extensions	<input checked="" type="checkbox"/> true	Valid selection.

Figure 8. Table for selecting customer-relevant properties to evaluate possible variants.

Legend

✓ The variant with the most check marks is the most suitable.

	TruBend 13 LB	TruBend 17 SX	TruBend 23 SX	TruBend 85 SB
cl _h 6 Axis	✓			
cl _h 13.0 Tons Bending Pressure	✓			✓
cl _h Bend Monitor Included	✓			✓
ch _h Gauge Finger Extensions	✓			✓

Figure 9. Representation of an evaluation of possible variants.

13LB has the most ticks and should be selected. The variant *TrueBend 85 SB* could also be chosen, but the property of the locating surfaces with the six axes is not fulfilled for this variant. Accordingly, the check mark is not set in the illustration.

Based on the modelled data connections, many other applications can be set up because of the architecture design. The model is a basis for implementing different views and advantages with little modelling effort. In this context, PFE, with its modelled definition of relations, is essential to consider all variants and their relations. PFE can be seen as a detailed view of the underlying product family regarding its variety and modularity. In this way, the overarching product line, which focuses on the external view of customer properties and their configuration, can be described in detail.

Using modelled system architectures results in further advantages, especially in the early dimensioning phase of development. Through the targeted use of cross-architecture calculation logic with the help of parametric diagrams, static parameters such as the mass of modules or variants can also be calculated, including the internal variety over the whole product family. In addition, state-dependent dynamic parameter calculations can be carried out with the help of simulations. Modelling tools such as the Cameo Systems Modeler have capabilities like the so-called Rollup Patterns, which use states controlled via a state machine for the recursive calculation of parameters. Thus, they can calculate dynamic changes within the system, such as power consumption or other flows (Berschik *et al.* 2022). This information can be used in the design to define margins at an early stage and to take them into account in further development.

By holistically linking several product variants of a product family in a system model, required margins can be verified against existing requirements at an early stage or reviewed during product generation development. Using such calculation tools is a great added value for the continued use of the architecture models. At this point, further research is needed, as the calculation logic must be better implemented in the existing models of the system architecture. This would allow for a better consideration of the variance of individual components in the overall system. The implementation is currently being further investigated.

4.2. Domain-specific models

In AS, collaboration-induced complexity can increase in addition to variant-induced complexity. Due to the increasing number of disciplines and domains involved, it is necessary to synchronise the drafts in the technical processes up to the design and to map them within the framework of a domain-specific model (Mensing & Schaefer 2015; Wu *et al.* 2022).

In the context of PFE, the processes up to the design stage are carried out multiple times and in parallel. Harmonising the designs between the variants and across the disciplines and domains creates a major action area with much synergy and savings potential (Gauss, Lacerda, & Miguel 2022).

To exploit the best possible potential, modularisation helps to reveal the synergies between the variants transparently and to address them in a targeted manner. The characteristics and properties of modularity help here, enabling structured reusability and interchangeability while securing the intended functions. Furthermore, however, the processes and the organisation must be harmonised (Colfer & Baldwin 2016; Yassine 2021). In PFE, this harmonisation is supported by various method

modules and tools. The goal is to derive a modular design harmonised for the product or system side, e.g., the Advanced System, and for the process and the organisation, i.e., the AE. This results in clearly defined interfaces, responsibilities, procedures, and processes in the maintenance and design of product families (Helmer, Yassine, & Meier 2010; Krause & Gebhardt 2023).

The process and the organisational collaboration can be represented by a collaboration-process-visualisation (CPV), as shown in Figure 10. The CPV enables different disciplines and domains in a development process to be represented transparently and collaboration to be derived from this. The CPV is based on the method-process-visualisation (MPV; Krause & Gebhardt 2023), extended by collaboration activities and critical indicators for collaboration-induced complexity. As shown in Figure 10, this modelling enables a more detailed analysis of the interaction of all participants in the AE process (Kolschoten *et al.* 2006; Mcharek *et al.* 2019).

In the *TruBend* product family example, these processes can be set up for, e.g., the systems engineer and the software architect. This results in two process streams containing mono-disciplinary and collaborative activities. The visualised interfaces and common references to peripherally built system models enable a targeted design of a common understanding with corresponding models in systems architecture design (Mensing & Schaefer 2015).

This makes it possible to create a common system model for developing the modular product family and not to promote autarkic domain-specific designs (Zuefle *et al.* 2022). In the example of the product family, collaborative decisions are thus made at the level of technical realisation and the corresponding functional description of the smart, networked back gauge, in which both the software for any business model and the constructive and electro-technical realisation of the assembly could be coordinated. This results in potential integration and preventive activities for error avoidance of multi-disciplinary structures (Fioriti *et al.* 2020; Rennpferdt & Krause 2020).

Depending on the use case and organisational structure, additional relevant stakeholders can be integrated into the CPV to develop the systems architecture designs. A prominent example of agile working is the product owner (Schmidt, Weiss, & Paetzold 2018).

The structure of the systems architecture designs of an AS and the preparation of the process of AE and its interfaces serve as the basis for multi-disciplinary modularisation in PFE. Here, the designed system components are assigned to the corresponding disciplines or domains and are also expanded by the scope of the domain-specific or discipline-specific components (Helmer, Yassine, & Meier 2010). This way, interactions can be analysed, interpreted and addressed from a process, organisational and product perspective. Tools such as modelling a Domain Harmonisation Chart (DHC) and allocation matrices between domains are used for this (Sosa, Eppinger, & Rowles 2004; Zuefle & Krause 2023). Currently, the model-based mapping of this methodological tool is a challenge in research, as the linkage must take place across three model domains.

By modelling the interactions and interfaces, suitable harmonised module sections can be derived and evaluated across various domains. To use these module sections for AE, they must be transferred back into the process and organisational domains, in which, for example, design sprints are established according to the harmonised assignment (Grimheden 2013). Furthermore, the activities in the

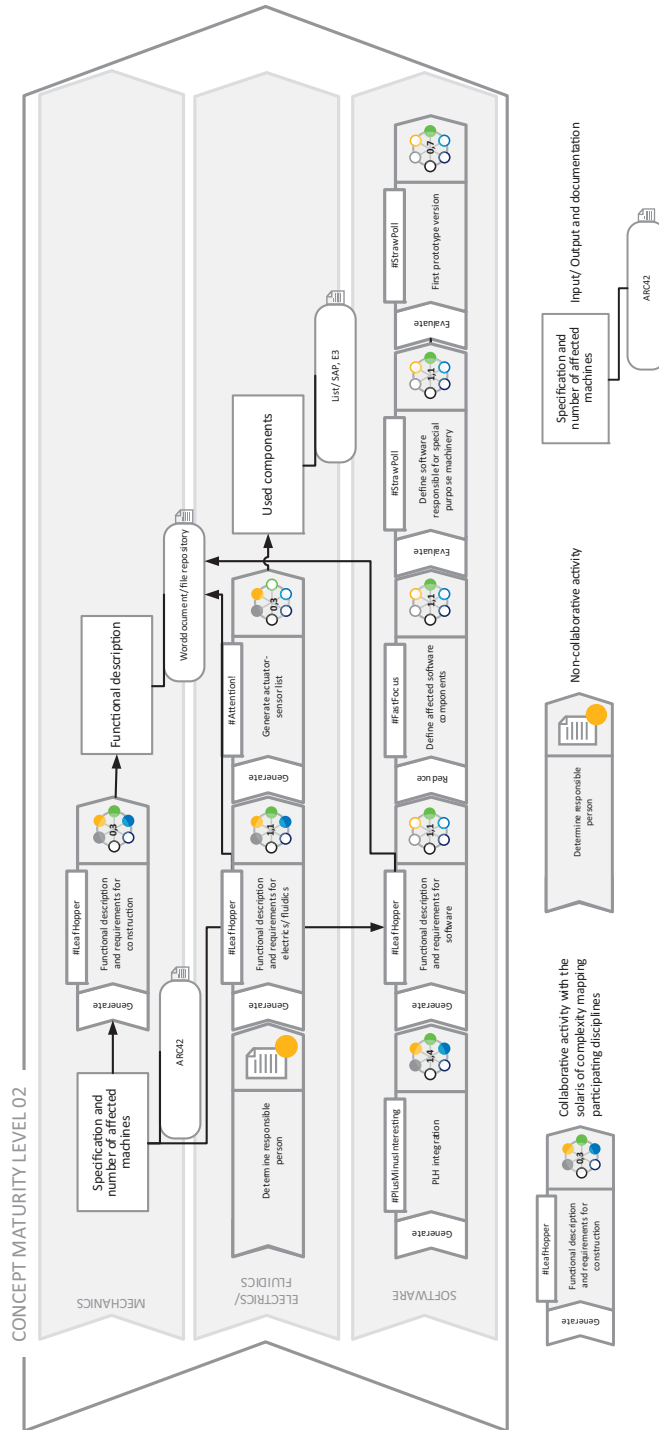


Figure 10. Extract from collaboration-process-visualisation (CPV) of a development process with three exemplary participants in MS Visio.

collaboration process can be adapted to the harmonised module sections, reducing the KPI-based assessment of the collaboration-induced complexity in the PV.

Using the *TruBend* product family as an example, the Domain Specific Models achieved harmonisation of the modules for the bending process and monitoring the bending angle here. The bending angle can be monitored and measured using various technical and software solutions, but the respective solution affects the implementation in the system architecture design. The identified collaboration between the systems engineer and the software architect in the case of kinematic measurement results in an interface for interaction in the engineering process. The implementation of the harmonisation in the DHC results in two modules, which, on the one hand, enable the bending process via the traversing and monitoring of the press beam, and on the other hand. This module allows for angle measurement using sensor-integrated tools and additional lasers. For both modules, corresponding collaborating module teams were formed, which take care of the subsystem development with defined interfaces.

4.3. Model-based testing and test-specifications

The validation and verification of systems, especially AS, is crucial to their certification. Without testing, the functional integrity of the system cannot be ensured (Walden *et al.* 2015). The test series is defined in the product-specific test plan. Test specifications are used to prescribe specific tests for subsystems or major components. Typically, test series are set up on the real system for verification and validation. If looking at these process steps from the perspective of model-based support, there are advantages if the test specifications use a model-based architecture design in the form of the system model (Pretschner & Philipps 2005, Utting & Legiard 2007, Weißleder 2010). Since they utilise the system's already used and modelled data and link relevant test cases to the predefined requirements (Schieferdecker & Hoffmann 2012), this reduces the documentation effort, as model elements can be standardised for the test procedure.

On the other hand, considering product families offers the possibility of reducing the test effort since modules that occur in several variants need to be tested only once (Krause & Gebhardt 2023). The final goal is to derive semi-automatic test scripts from the model-based test specifications that can holistically reduce the execution and complexity of large-scale test series. For test procedures and evaluations to be created semi-automatically, depending on what tests have already been conducted with similar variants, there is a need for a comprehensive definition and modelling of test requirements, boundary conditions and, of course, the product family itself. Regarding the occurring variance within product families, the presented model-based PFE can lay the ground for further implementations regarding the strategic testing of its variants. Ideally, outcomes for different subsystems or variants are fed back into the system model to allow for an easy overview of already conducted and pending tests when focusing on another variant. The implementation of this approach is part of current research.

4.4. Retrofit and maintenance—challenging processes during the product usage

The previously described aspects and approaches focus on different parts of product creation and can be connected to other phases within the V-Modell

(Figure 3). While the main engineering activities are mostly finished at that point, most of the product's life, its usage phase, begins. Yet, the presented aspects of PFE remain relevant, especially for long-living products and products with rigorous safety and maintenance requirements.

Staying with the example of the *TruBend* bending machine product family, it is developed as a product family but manufactured as tailored variants for each customer. Even if a single customer orders three identical variants, while being ideally identical at the point of delivery, these three instances of the product will start to deviate from the documented and initial state.

In one instance of the product, a collision of the bending tool may occur due to improper handling. The second instance is being used more heavily or even above its limits and has worn out rails after some time. Because of a fabrication failure in a vendor part, the hydraulic ducts of the third instance start to leak. All product instances are being repaired during regular maintenance, albeit they remain different. The collision damage has impaired the functionality, as even with a replacement part, the machine needs to cope with still warped structural parts. This can be compensated in software but limits the speed of operation. The worn-out rail of the second product instance is being replaced by a more durable one. All three instances have also received a crucial software update.

As a result, deviations between the product instances occur over time, even if they once were identical variants. Defects, wear and tear, or replacements as part of smaller maintenance works cause increasingly more deviations of the specific product instances than the ideal product variant. Thus, the occurring variability is not solely determined by the planned variance within the product family anymore but also by the emerging deviance, the deviation of the actual product from its once ideal state.

In this situation, an upgrade of all machines from four- to six-axe operations is planned in coordination between the customer and the original manufacturer. While the generally possible upgrade solutions are defined by the product family architecture and its possible variants, there may be limitations to the implementation because of individual deviations. However, to identify these limitations, information about the actual state of all instances first needs to be gathered to evaluate the actual upgrade possibilities. This overall situation is visualised in Figure 11.

Exact knowledge of the product family greatly reduces the challenge of the variability, even in this later phase. The knowledge of the variance allows the reduction of the overall faced variability to the deviance originating from the usage.

Aviation is another application where this variability, combined with variance and deviance, plays an important role. A closer look reveals that the ongoing maintenance works and, notably, the regularly occurring retrofit face a challenging amount of variability within the targeted aircraft. The cabin retrofit can be described as an aircraft modification, during which parts of the passenger cabin are replaced to face wear and tear and to update the cabin to new layouts. Hence, a new cabin is being developed for an already existing aircraft, manufactured and finally installed into the specific airframe (Moenck *et al.* 2022). This process requires solid planning data regarding the airframe early on, as each aircraft is unique, and information specific to the very aircraft the cabin is being planned for is required (Laukotka & Krause 2023). Fortunately, aviation is actually characterised by product families. The famous *A320 family* consists of the basis-variants

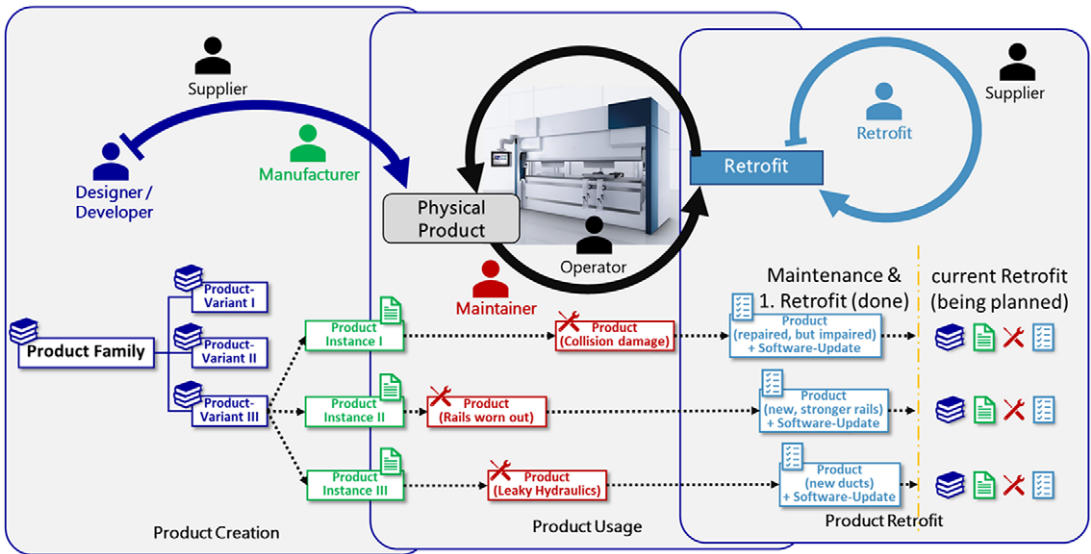


Figure 11. Variability of products and their respective documents when facing maintenance and retrofit.

A318, A319, A320 and A321, which are mainly differentiated by their length and, thus, capacity. However, most aspects of these aircraft are the same. This existing variance aligns with the previously described and conventional variability within product families. Each of these product variants is manufactured multiple times, and thus, multiple instances with identical layouts and states are delivered to the airline.

Again, with time, deviations between the once identical instances arise. Despite there being documentation for every planned modification to the aircraft, they are usually limited to the respective area of the aircraft the modification was done to and separated by the different affected domains (*electric, structure, water/waste, etcetera*). As a result, the available documentation is fragmented and must be manually stitched together to allow for a somewhat realistic insight. This poses a real challenge in determining the aircraft’s actual state, as required for planning and developing the new interior during the cabin retrofit (also see Figure 11). As also within the earlier stages of the product life, solid documentation of the product family will help to cope with this challenge, as it greatly reduces the variability by clearly structuring and defining the variance and, thus, easing the handling of information for the engineers handling the data for the retrofit (Laukotka, Hanna, & Krause 2021). At this, two typical scenarios must be differentiated: The retrofit performed by the original manufacturer or a third party.

In case the manufacturer performs the retrofit, like the described example of the bending machine, detailed as-planned information about the product and the family structure is available. Tracking the state of the products after their production corresponds to the current concepts of Digital Twins as presented, for example, by Stark *et al.* (2020). As the amount of information rises when considering product families, a model-based approach can again assist with handling the data (Laukotka & Krause 2023). Because in this scenario, the original manufacturer does the retrofit, ideally, these models already exist from the phase of product

creation, as described in earlier sections. Thorough modelling and availability of information enable the switch between variants during the retrofit or even the retrospective implementation of new variants into the product family. Much information about a specific aircraft can be easily obtained by accessing information about another variant or the product family. The available information regarding the product family makes the variance easily handled. However, the deviance, albeit minor, needs to be identified. Based on the documentation described above, this task can be accomplished, especially with the overall insights the manufacturer has generally available.

The information deficit is more significant if a third party performs the retrofit, like specialised maintenance- and retrofit organisations. The as-planned information about the product, especially the product family, is unavailable, and a Digital Twin of the product would have to be initialised mid-life, based on the limited information (Moenck *et al.* 2022). Because of the similarity of aircraft and their underlying family structure, besides the top-down approach of the developer and manufacturer, there is a bottom-up approach of iteratively gaining the required information. Some information can be classified as generic and applicable for all aircraft (of the same type or fleet), while others are specific to the aircraft at hand (Laukotka & Krause 2023). A thorough modelling of information on these different levels, iteratively gained from generally available data and previous retrofits, allows for compiling enough knowledge to plan and perform the retrofit. Compared to the first scenario, this depicts a much bigger task. The vast number of documents required for the certification of each individual aircraft poses a challenge and a chance at the same time in this regard.

Both examples of products and the depicted scenarios of the retrofit show how the occurring variability can be handled at least partly by considering the underlying product family. Model-based approaches, created top-down during the product creation or bottom-up mid-life can once again help with this aspect of PFE.

5. Conclusion

Variability is found throughout the life cycle of a product, especially in product families. Managing these product families becomes increasingly complex the larger the systems under consideration become. ASE offers methods, approaches and experience to cope with these complex boundaries in those systems.

In this contribution, the importance of dealing with variability throughout the life cycle of a product family and considering its context was presented using the example of a sheet metal bending machine product family. For this purpose, individual aspects in different product life cycle phases were selected, ranging from essential steps during product development to aspects during product usage. The aspects and examples presented are selected directly from current and ongoing research.

The modelling of crucial information, inspired by the progress of model-based approaches found in SE and MBSE, is one fundamental basis of most of the presented research.

Different examples were described throughout this work by referring to the formulated research question, how the explicit and strategic consideration of

product families and their variability in the context of ASE can improve the development and its further product life cycle.

The advantages of a consistent model-based implementation of the system architecture of a product family in the context of ASE were explicitly addressed. It was shown how, based on the modelled system architecture, different views could be derived that offer additional value with comparatively small modelling efforts. It was shown that PFE focuses on describing and managing variant-induced complexity and accordingly focuses on minimising variety, for example, through modular structures. This contrasts with the PLE, which focuses on mapping and configuring different variants based on predefined domain assets. PFE focuses on the description of these domain assets under variety-induced complexity aspects. It can, therefore, be located within a product line while considering the underlying product families. In addition, applying domain-specific models was considered, ensuring collaboration within AS' development and encouraging transparency. A short excursus was given on model-based documentation of test specifications based on a system model, and the topic of retrofits in the usage phase of product families was examined.

A challenge that remains, independent from the scenario and in general for most data handling approaches that reach into the usage phase with its multiple stakeholders, is the maintenance of the documentation. While updating models during the product creation can be done comparatively quickly because the product, its information and the respective documentation follow a clearly defined process and usually stay within a single or few but well-connected stakeholders, this cannot be said for the usage phase. If no coherent models are transferred from the creation to the usage, they can be created bottom-up as presented above. Yet, they still need to be updated for an ongoing benefit. Yet, this challenge is not unique to model-based approaches. The concept of Digital Twins can often be found applied to manufacturing facilities with assets in close proximity. However, the application of the concept to products with a high quantity and diverse customers faces a similar challenge. Research remains to be done to improve this situation, aside from the recommendation to keep the models as up-to-date as possible.

This contribution demonstrated that an explicit and strategic view can establish an in-depth understanding of product families and their variability. It enables the challenges that ASE is dealing with to be addressed from an additional point of view. PFE enhances the current approach of ASE by focusing on the development level of variability and the potential of modularisation through product families.

6. Outlook

This work has presented different research aspects to cope with variability in AS along the life cycle. This PFE goes back to different challenges during the research work within multiple projects and with different industry partners. Despite that, they are facing comparable challenges and have similar approaches. However, despite their similarity, they mostly stand alone for now. An advisable next step to ensure compatibility between the different aspects is to create a holistic framework that allows for consistent data handling.

While multiple aspects were presented in this work, current research considers even more applications of the presented approach of PFE. For example, and similarly to the retrofit, complementary services accompany the product mainly during its

usage phase. In the case of product families, these product service systems can greatly benefit from exact knowledge about the variability. A thorough PFE from the get-go is key to a performant provision of the different service aspects and considering the occurring variability. More work is needed regarding the product architecture, especially the parametric description of product families enabling recursive calculations in early design phases. This would also enhance the possible automatic definition of test specifications which currently depicts an area with only a few research and implementations, particularly for product families in AS.

More research must be done further to increase the potential of PFE in ASE. Focusing on practical implementations and case studies would coincidentally increase the acceptance and prominence of these approaches in the community and the industry.

Acknowledgements

Publishing fees are supported by the Funding Programme Open Access Publishing of Hamburg University of Technology (TUHH).

References

- Anacker, H., Dumitrescu, R., Kharatyan, A. & Lipsmeier, A. 2020 Pattern based systems engineering - Application of solution patterns in the design of intelligent technical systems. In *Proceedings of the Design Society: DESIGN Conference*, pp.1195–1204. <https://doi.org/10.1017/dsd.2020.107>.
- Arora, R. & Raja, M. 2016 Advanced Engineering Materials and Their Applications in Aerospace Industry. *Materials Science and Engineering Journal* 55 (4), 189–206.
- Aßmann, U., Zschaler, S. & Wagner, G. 2006 Ontologies, meta-models, and the model-driven paradigm. In *Ontologies for Software Engineering and Software Technology*, pp. 249–273. Springer. https://doi.org/10.1007/3-540-34518-3_9.
- Berschik, M. C., Oidtmann, N., Laukotka, F. N., Brahm, M. & Krause D. 2022 Model-based power budget analysis of satellites using SysML Rollup Pattern. In *Gesellschaft für Systems Engineering e.V* (eds. Koch, W., Wilke, D., Dreiseitel, S. & Kaffenberger, R.), pp. 75–79. Tag Des Systems Engineering.
- Berschik, M. C., Schumacher, T., Laukotka, F. N., Krause, D. & Inkermann, D. 2023 MBSE within the engineering design community – an exploratory study. *Proceedings of the Design Society* 3, 2595–2604. <https://doi.org/10.1017/pds.2023.260>.
- Blecker, T. & Abdelkafi, N. 2006 Complexity and variety in mass customisation systems: analysis and recommendations. *Management Decision* 44 (7), 908–929.
- Brown, C. & Smith, M. 2019 The role of advanced engineering in sustainable product design and manufacturing. *Journal of Cleaner Production* 78 (5), 225–242.
- Colfer, L. J. & Baldwin, C. Y. 2016 The mirroring hypothesis: theory, evidence, and exceptions. *Industrial and Corporate Change* 25 (5), 709–738.
- Colletti, R. A., Qamar, A., Nuesch, S. P. & Paredis, C. J. J. 2020 Best practice patterns for variant modeling of activities in model-based systems engineering. *IEEE Systems Journal* 14 (3), 4165–4175. <https://doi.org/10.1109/JSYST.2019.2939246>.
- Delligatti, L. 2014 *SysML Distilled. A Brief Guide to the Systems Modeling Language*. Addison-Wesley.
- Eigner, M. & Gilz, T. 2012 Proposal for functional product description as part of a PLM solution in interdisciplinary product development. In *International Design Conference (DESIGN)*, Dubrovnik, Croatia.

- ElMaraghy, W., ElMaraghy, H., Tomiyama, T. & Monostori, L. 2012 Complexity in engineering design and manufacturing. *CIRP Annals* **61** (2), 793–814.
- Fioriti, M., Boggero, L., Prakasha, P. S., Mirzoyan, A., Aigner, B. & Anisimov, K. 2020 Multi-disciplinary aircraft integration within a collaborative and distributed design framework using the AGILE paradigm. *Progress in Aerospace Sciences* **119**, 100648.
- Friedenthal, S., Moore, A. & Steiner, R. 2015 *R: A Practical Guide to SysML - The System Modeling Language*, 3rd edn. Elsevier. <https://doi.org/10.1016/C2013-0-14457-1>.
- Gauss, L., Lacerda, D. P. & Miguel, P. A. C. 2022 Market-driven modularity: Design method developed under a design science paradigm. *International Journal of Production Economics* **246**.
- Grimheden, M. 2013 Can agile methods enhance mechatronics design education? *Mechatronics* **23** (8), 967–973.
- Harlou, U. 2006 Developing product families based on architectures: Contribution to a theory of product families. Dissertation, Denmark: Technical University of Denmark.
- Haugen, Ø., Wąsowski, A. & Czarnecki, K. 2012 CVL: common variability language. In *Proceedings of the 16th International Software Product Line Conference, New York, USA*.
- He, L., Chen, W. & Lee, H. 2018 Emerging trends in advanced systems and their impact on industry 4.0. *International Journal of Advanced Manufacturing Technology* **37** (1), 78–95.
- Hehenberger, P., Vogel-Heuser, B., Bradley, D., Eynard, B., Tomiyama, T. & Achiche, S. 2016 Design, modelling, simulation and integration of cyber physical systems: Methods and applications. *Computers in Industry* **82**, 273–289.
- Helmer, R., Yassine, A. & Meier, C. 2010 Systematic module and interface definition using component design structure matrix. *Journal of Engineering Design* **21** (6), 647–675.
- Holt, J., Perry, S. A. & Brownsword, M. 2012 *Model-Based Requirements Engineering: IET Professional Applications of Computing Series V9*. Institution of Engineering and Technology.
- Hummell, J. & Hause, M. 2015 Model-based product line engineering - enabling product families with variants. In *2015 IEEE Aerospace Conference, Big Sky, MT, USA*, pp. 1–8. IEEE. <https://doi.org/10.1109/AERO.2015.7119108>.
- INCOSE (Ed.) 2007 INCOSE Systems Engineering Vision 2020. INCOSE - Technical Operations INCOSE-TP-2004-004-02). Available online at https://sdincose.org/wp-content/uploads/2011/12/SEVision2020_20071003_v2_03.pdf.
- Inkermann, D. 2021 Shaping method ecosystems - Structured implementation of systems engineering in industrial practice. *Proceedings of the Design Society* **1**, 2641–2650. <https://doi.org/10.1017/pds.2021.525>.
- Isermann, R. 2008 *Mechatronische Systeme: Grundlagen*. <https://doi.org/10.1007/978-3-540-32512-3>.
- ISO/IEC/IEEE (Ed.) 2015 International Standard ISO/IEC/IEEE 15288: Systems and software engineering - System life cycle processes, 35.080 Software.
- ISO/IEC/IEEE (Ed.) 2021 International Standard ISO/IEC/IEEE 26580, Software and systems engineering - Methods and tools for the feature-based approach to software and systems product line engineering, 35.080 Software.
- Japs, S. 2020 Security & safety by model-based requirements engineering. In *2020 IEEE 28th International Requirements Engineering Conference (RE), Zurich, Switzerland*, pp. 422–427. IEEE. <https://doi.org/10.1109/RE48521.2020.00062>.
- Kim, J., Park, S. & Chang, Y. 2021 Application of advanced systems in autonomous vehicles: challenges and prospects. *Journal of Intelligent Transportation Systems* **32** (2), 201–218.

- Kleiner, S. & Kramer, C.** 2013 Model based design with systems engineering based on RFLP using V6. In *CIRP Design Conference. Volume. Smart Product Engineering* (eds. Abramovici, M. & Stark, R.). Springer. https://doi.org/10.1007/978-3-642-30817-8_10.
- Kolfschoten, G. L., Briggs, R. O., de Vreede, G. J., Jacobs, P. H. M. & Appelman, J. H.** 2006 A conceptual foundation of the think let concept for Collaboration Engineering. *International Journal of Human-Computer Studies* **64** (7), 611–621.
- Krause, D. & Gebhardt, N.** 2023 *Methodical Development of Modular Product Families - Developing High Product Diversity in a Manageable Way*. Springer. <https://doi.org/10.1007/978-3-662-65680-8>.
- Laukotka, F., Hanna, M. & Krause, D.** 2021 Digital twins of product families in aviation based on an MBSE-assisted approach. *Procedia CIRP* **100**, 684–689. <https://doi.org/10.1016/j.procir.2021.05.144>.
- Laukotka, F. N. & Krause, D.** 2023 Supporting digital twins for the retrofit in aviation by a model-driven data handling. *Systems* **11**, 142. <https://doi.org/10.3390/systems11030142>.
- Liu, Y., Smith, J. & Zhang, Q.** 2017 Advancements in advanced systems engineering for complex industrial applications. *Journal of Advanced Engineering Studies* **13** (2), 45–62.
- Martin, J. N.** 1997 *Systems Engineering Guidebook*. CRC Press.
- Mcharek, M., Azib, T., Larouci, C. & Hammadi, M.** 2019 Collaboration and multi-disciplinary design optimisation for mechatronic systems. In *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, Vol. **1**, pp. 624–629. IEEE.
- Mensing, B. & Schaefer, I.** 2015 A methodology for hierarchical multi-disciplinary modeling and analysis of mechatronic systems. In *Tagungsband - Dagstuhl-Workshop MBES 2015: Modellbasierte Entwicklung Eingebetteter Systeme XI*.
- Mertens, K. G., Rennpferdt, C., Greve, E., Krause, D. & Meyer, M.** 2022 Reviewing the intellectual structure of product modularisation: Toward a common view and future research agenda. *Journal of Product Innovation Management* **40**, 86–119.
- Metzger, A. & Pohl, K.** 2014 Software product line engineering and variability management: achievements and challenges. In *Future of Software Engineering Proceedings (FOSE 2014)*, pp. 70–84. Association for Computing Machinery. <https://doi.org/10.1145/2593882.2593888>.
- Meyer, M. H. & Lehnerd, A. P.** 1997 *The Power of Product Platforms – Building Value and Cost Leadership*. Free Press. [https://doi.org/10.1016/S0737-6782\(97\)80157-9](https://doi.org/10.1016/S0737-6782(97)80157-9).
- Moenck, K. H. W., Laukotka, F. N., Krause, D. & Schüppstuhl, T.** 2022 Digital twins of existing long-living assets: reverse instantiation of the mid-life twin. In *Proceedings of the 33rd Symposium Design for X (DFX2022), Hamburg, Germany*. <https://doi.org/10.35199/dfx2022.20>.
- Morgan, M., Holzer, T. & Eveleigh, T.** 2021 Synergizing model-based systems engineering, modularity, and software container concepts to manage obsolescence. *Systems Engineering* **24** (5), 369–380.
- Mortensen, N. H.** 1999 Design modelling in a designer's workbench—contribution to a design language. Dissertation, Denmark: Technical University of Denmark.
- Pirmoradi, Z. & Wang, G. G.** 2011 Recent advancements in product family design and platform-based product development: A literature review. In *Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pp. 1041–1055.
- Pohl, K., Böckle, G. & Linden, F.** 2005 *Software Product Line Engineering. Foundations, Principles, and Techniques, with 10 Tables*. Springer Verlag.

- Pohl, K. & Rupp, C.** 2011 *Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam - Foundation Level - IREB Compliant* (1st edn). Rocky Nook.
- Pretschner, A. & Philipps, J.** 2005 Methodological issues in model-based testing. In *Model-Based Testing of Reactive Systems - Advanced Lectures* (eds. Broy, M., et al.), pp. 281–291. Springer.
- Rennpferdt, C. & Krause, D.** 2020 Towards a framework for the design of variety-oriented product-service systems. *Proceedings of the Design Society: Design Conference 1*, 1345–1354. <https://doi.org/10.1017/dsd.2020.108>.
- Robertson, D. & Ulrich, K.** 1998 Planning for product platforms. *Sloan Management Review* 39 (4), 19–31.
- Rupp, M. A.** 1988 *Produkt-, Markt-Strategien – Handbuch zur marktsicheren Produkt und Sortimentsplanung in Klein- und Mittelunternehmungen der Investitionsgüterindustrie*. Verlag Industrielle Organisation.
- Salvador, F.** 2007 Toward a product system modularity construct: literature review and reconceptualization. *IEEE Transactions on Engineering Management* 54(2), 219–240.
- Schieferdecker, I. & Hoffmann, A.** 2012 Model-based testing, *IEEE Software* 291, 14–18.
- Schmidt, T. S., Weiss, S. & Paetzold, K.** 2018 *Agile Development of Physical Products: An Empirical Study about Motivations, Potentials and Applicability*. Universitätsbibliothek der Universität der Bundeswehr München.
- Simpson, T. W., Siddique, Z. & Jiao, J.** 2006 *Product Platform and Product Family Design: Methods and Applications*. Springer Verlag. <https://doi.org/10.1007/0-387-29197-0>.
- Sosa, M. E., Eppinger, S. D. & Rowles, C. M.** 2004 The misalignment of product architecture and organizational structure in complex product development. *Management Science* 50(12), 1674–1689.
- Stark, R., Anderl, R., Thoben, K.-D. & Wartzack, S.** 2020 WiGeP-Positionspapier: “Digitaler Zwilling”. *Zeitschrift für wirtschaftlichen Fabrikbetrieb* 115, 47–50. <https://doi.org/10.3139/104.112311>.
- Tomiyama, T., Lutters, E., Stark, R. & Abramovici, M.** 2019 Development capabilities for smart products. *CIRP Annals* 68(2), 727–750.
- Tukker, A.** 2015 Product services for a resource-efficient and circular economy – a review. *Journal of Cleaner Production* 97, 76–91. <https://doi.org/10.1016/j.jclepro.2013.11.049>.
- Ulrich, K. T.** 1995 The role of product architecture in the manufacturing firm. *Research Policy* 24(3), 419–440. [https://doi.org/10.1016/0048-7333\(94\)00775-3](https://doi.org/10.1016/0048-7333(94)00775-3).
- Utting, M. & Legeard, B.** 2007 *Practical Model-Based Testing*. Morgan Kaufmann.
- VDI/VDE (Ed.)** 2019 Richtlinie VDI/VDE 2221: Design of technical products and systems Model of product design.
- VDI/VDE (Ed.)** 2021 Richtlinie VDI/VDE 2206: Entwicklung mechatronischer und cyber-physischer Systeme.
- Walden, D. D., Roedler, G. J., Forsberg, K., Hamelin, R. D. & Shortell, T. M. (Eds.)** 2015 *Systems engineering handbook. A guide for system life cycle processes and activities. In INCOSE-TP-2003-002-04, International Council on Systems Engineering*, 4th edn. Wiley.
- Weber, C. & Husung, S.** 2016 Solution Patterns – Their role in innovation, practice and education. In *Proceedings of the Design Society: DESIGN Conference*, pp. 99–108.
- Wegmann, M., Johnson, L. & Anderson, S.** 2020 Integrating human factors into advanced systems engineering: Challenges and opportunities. *International Journal of Systems Engineering* 25 (4), 321–339.
- Weilkiens, T.** 2016 Variant modeling with SysML. Fredesdorf: MBSE4U.

- Weißleder, S.** 2010 *Test Models and Coverage Criteria for Automatic Model-Based Test Generation with UML State Machines*, Dissertation, Humboldt-Universität.
- Wu, Y., Zhou, L., Zheng, P., Sun, Y. & Zhang, K.** 2022 A digital twin-based multi-disciplinary collaborative design approach for complex engineering product development. *Advanced Engineering Informatics* **52**, 101635. <https://doi.org/10.1016/j.aei.2022.101635>.
- Wymore, A. W.** 1993 *Model-Based Systems Engineering*. CRC Press.
- Yassine, A. A.** 2021 Managing the development of complex product systems: An integrative literature review. *IEEE Transactions on Engineering Management* **68**(6), 1619–1636.
- Zuefle, M. & Krause, D.** 2023 Multi-disciplinary product design and modularization – Concept introduction of the Domain Harmonization Chart (DHC). *Procedia CIRP* **119**, 938–943.
- Zuefle, M., Muschik, S., Bursac, N. & Krause, D.** 2022 Coping asynchronous modular product design by modelling a systems-in-system. In *17th International Design Conference*.