# Enhancing Collaborative Modular Product Development: Interface Allocation and Associated Responsibilities

Marc Zuefle[1], Jan Küchenhof[1], Markus Christian Berschik[1], Dieter Krause[1]

[1]Hamburg University of Technology (TUHH), Institute of Product Development and Mechanical Engineering Design (PKT), Denickestraße 17, 21071 Hamburg, Germany

**Abstract:** For meeting the dynamic demands of customers on a product, development increasingly relies on expertise from a diverse array of disciplines and domains. Only this collaboration makes it possible to provide comprehensive and practical approaches to solutions for various challenges. However, in addition to variety-induced complexity, this adds complexity due to increased collaboration. Particularly in developing modular product families, this circumstance leads to an expanded consideration of collaboration in subsystems, enabling effective product-family-system development. Collaboration in modular product families leads to a necessary integration of the consideration of interfaces' dependencies, responsibilities, and specifications. This paper presents an approach to identify and manage interface responsibilities in multi-disciplinary modular product-family systems using product and organization view matrices. Results from previous studies are put into a processual context to coordinate the collaboration considered. A matrix-based solution on product and allocated organizational elements improves collaboration.

*Keywords: Modularization, Collaboration, Product Family, Multi-Disciplinarity, Interfaces*

## 1 Introduction

Due to the increasing development of connected and intelligent products, the structuring of product architectures is becoming much more extensive than before (Mertens *et al.*, 2022). Many products have an expanding proportion of additional software-specific components due to the transition to cyber-physical systems (Hehenberger *et al.*, 2016). The ratio of the electrical and other discipline- and domain-specific components is also increasingly important. Thus, these products are increasingly evolving into multi-disciplinary (e.g., mechatronic) systems, and the increasing interdependence also increases the resulting collaboration-induced complexity (ElMaraghy *et al.*, 2012; Tomiyama *et al.*, 2019). The collaboration-induced complexity thus represents, in addition to the variety-induced complexity, a further level of complexity that must be mastered in developing such systems (Yassine, 2021). Developing modular product families represents a valid and effective way of mastering variety-induced complexity at the product architecture level. However, to master the collaboration-induced complexity that arises in developing multi-disciplinary systems, new approaches have to be elaborated (Beck *et al.*, 2001; Cadavid *et al.*, 2021). To master collaboration-induced complexity in modular architectures, the responsibilities in those architectures must be addressed precisely and at which level in the system they are. Those include, on the one hand, the responsibilities for the design and realization of the subsystems, modules, or components and the responsibility and competence for defining and designing the interfaces in and outside a module (Salvador, 2007; Krause and Gebhardt, 2023). The systematic analysis and design of interfaces in multi-disciplinary systems enable the targeted derivation of collaboration in modular structures, adding value to the multi-disciplinary design of modules and their overall systems. In this elaboration, this circumstance in multi-disciplinary modular product families is considered in more detail by analyzing the given research background in Section 2 and a short insight into the methodological approach and its previous studies in Section 3. Based on this work, collaboration and its impact on responsibilities and competencies in a modular architecture are analyzed in Section 4. Subsequently, Section 5 shows an application of the concept by treating an exemplary system of a robot vacuum cleaner accordingly. Finally, Section 6 critically examines the work presented here and provides an outlook on subsequent studies.

## 2 Research Background

This elaboration deals with collaboration in modular multi-disciplinary systems. For this purpose, the research background focuses on the following areas of relevance. On the one hand, the development of mechatronic systems, which represent a specific form of multi-disciplinary systems, and their collaboration will be discussed. On the other hand, the area of modular product families is dealt with, providing the framework for considering a modular system. Subsequently, a brief insight into research on collaboration is given in the Research Background.

### 2.1 Development of Mechatronic Products

In this paper, the development of multi-disciplinary systems is equated with the development of mechatronic systems. Those Systems require increasing interaction of different development disciplines, such as mechanics, electronics, and software. In this case, disciplines are specific fields of engineering based on, e.g., VDI 2206, but in reality, they can vary from organization to organization. Other multi-disciplinary systems, e.g., would be cyber-physical systems and cybertronic elements (Eigner *et al.*, 2014; Küchenhof *et al.*, 2022). The V-model, therefore, is often used as a development framework for developing mechatronic systems. This model helps to capture the requirements and to create discipline-specific system designs on this basis. Subsequently, the discipline-specific designs are integrated into the overall system via the right-hand side of the V-model and verified and validated via each integration state (Graessler *et al.*, 2018). In the context of mechatronic systems development, the V-model is linked to the higher-level life cycle of the system, from which the requirements for development and the functionality for usage emerge (Eigner *et al.*, 2014). These requirements are specified into functions, logic, and discipline-specific designs with discipline-specific components to implement the requirements effectively (Cabrera *et al.*, 2008; Graessler *et al.*, 2018). The discipline-specific design phases form the basis for the architecture of the resulting system and thus also influence the development of a modular Cyber Physical Product Family (CPPF) (Küchenhof et al., 2022). As Figure 1 illustrates, there is a jump from the abstract and generic level to the mono-discipline components level, addressing collaboration and behavior in a multi-disciplinary system (Cabrera *et al.*, 2008).

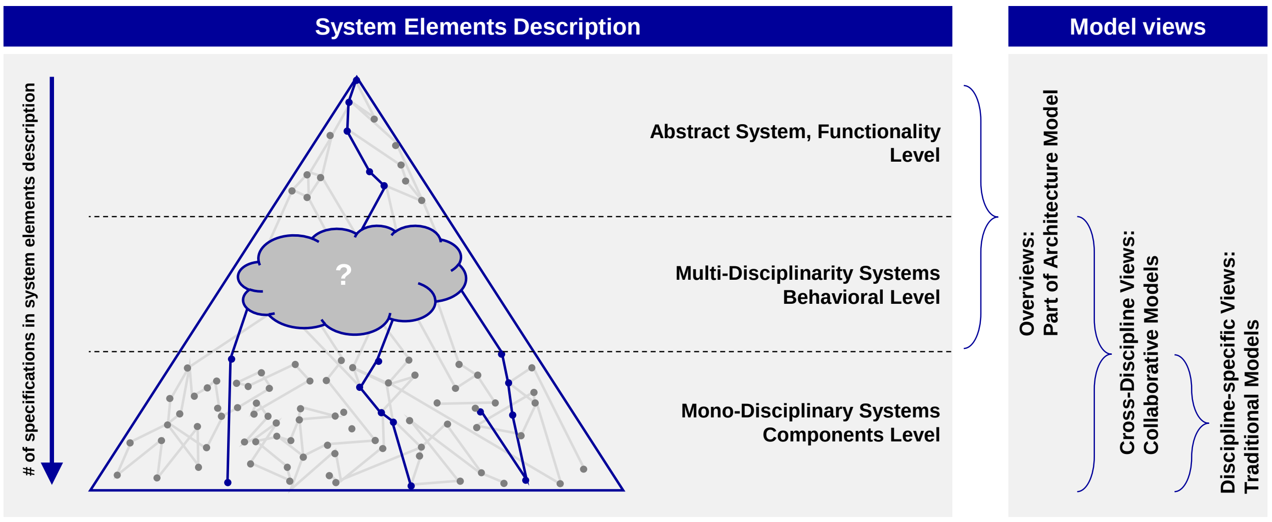
System Elements Description
Model views
# of specifications in system elements description
?
Abstract System, Functionality Level
Multi-Disciplinarity Systems Behavioral Level
Mono-Disciplinary Systems Components Level
Overviews: Part of Architecture Model
Cross-Discipline Views: Collaborative Models
Discipline-specific Views: Traditional Models

Figure 1. System descriptions from abstract to mono-disciplinary components in multi-disciplinary systems adapted from Cabrera *et al.* (2008)

## 2.2 Development of Modular Product Families

In addition to mechanics, the relevance of other development disciplines is increasing in developing new products and product families (Mertens *et al.*, 2022). The rising variety-induced complexity of the emerging mechatronic systems due to new components and increasing digital shares, which require both electronics and software, thus results in new interactions and challenges for product development. To cope with the high variety-related complexity of such systems, the development of modular product families is a possible solution (Simpson *et al.*, 2014; Otto *et al.*, 2016).

Generally, the approaches and methods for developing modular product families can be divided according to technical-functional and product-strategic objectives. For technical-functional structuring, *Design Structure Matrix (DSM) is* mentioned (Eppinger *et al.*, 2014). One exemplary product-strategic approach is *Modular Function Deployment (MFD),* using module drivers over diverse life phases for modularization (Erixon, 1998). In addition, there are also approaches in which both objectives are integrated, such as the *Product Family Master Plan* (Simpson *et al.*, 2014) or the *Integrated PKT-Approach (Krause et al., 2014).* The *Integrated PKT-Approach* is used to investigate the cross-disciplinary product architecture discussed due to the provided method with which the product architecture can be optimized regarding the relationship between the external and the internal variety. Furthermore, analogous to the RFLP-Approach, the method maps interactions across development levels in a branch & bound and divide & conquer-like manner (Eigner *et al.*, 2014). The method ensures that the existing product architecture is first analyzed and designed for necessary variety before modularization (Krause *et al.*, 2014).

A matrix-based approach like the *DSM* enables much potential for integrating the organizational view and the relevant interfaces for collaboration into modularization. Those Approaches are used in various application areas, including designing, analyzing, and structuring products, processes, and organizational tasks. A *DSM* is a square intra-domain matrix representing system elements such as product components, process steps, or organizational structures in the diagonal cells and couplings in the non-diagonal entries (Browning and Yassine, 2016). Relationships and interactions between and within domains can be further analyzed in *Multi-Domain Matrices (MDM)* and *Domain-Mapping-Matrices (DMM)* (Danilovic and Browning, 2007). The matrix-based approach allows linking relationships within and between different views, such as product structure in terms of *DSM* and product architecture, with an organizational view in terms of *MDM* and *DMM*. Those possibilities enable the integrated analysis of product, process, and organization for coping with complexity in product architecture (Sosa *et al.*, 2004).

Different elaborations can be considered in the context of the multi-disciplinary development of modular product families. The *Mechatronic Modularization (van Beek et al., 2010)*, the *Mechanics, Electronics, and Software Architecture (MESA)* (Askhøj *et al.*, 2021), and the *Module Harmonization Chart (MHC)* (Zuefle and Krause, 2023) are exemplarily mentioned. All these approaches consider interfaces in a modular product architecture. Askhøj et al. focus on the development disciplines as the organizational domain and investigates logical interactions. Van Beek looks at the functional interfaces of the product domain. In addition, Zuefle et al. aim to consider structural and organizational interfaces, including module drivers for product-strategic modularization across disciplines and domains.

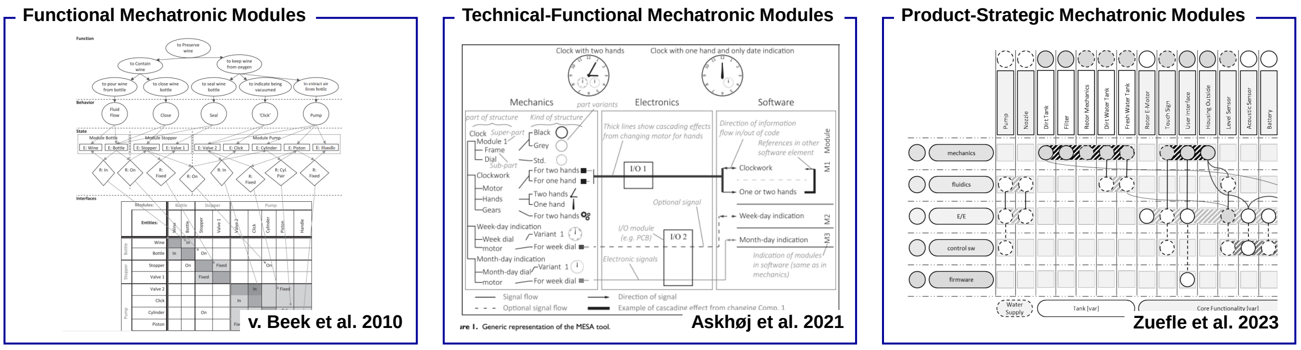
Functional Mechatronic Modules
v. Beek et al. 2010
Technical-Functional Mechatronic Modules
Clock with two hands
Clock with one hand and only date indication
Mechanics
Electronics
Software
Askhøj et al. 2021
Product-Strategic Mechatronic Modules
mechanics
fluidics
E/E
control sw
firmware
Zuefle et al. 2023

Figure 2. Selection from three approaches for the modularization of mechatronic systems

Based on the presented approaches, extending the DSM to an MDM involving multiple development disciplines can be a value-adding approach to modularization (Lambe and Martins, 2012). However, a detailed consideration of the multi-disciplinary interface design on the product side and its effects on the organizational design, such as responsibilities for modules, activities, and frequency of exchange, is missing in the literature and requires detailed consideration.

The specification of interfaces is already viewed in some concerns in literature, also in respect to matrix-based approaches (Mikkola and Gassmann, 2003; Helmer *et al.*, 2010). Also organizational interfaces in design team interaction are implemented in matrix-based approaches for investigating internal and external system design interfaces (Sosa *et al.*, 2003). Furthermore, the standardization or a targeted specification of interfaces is often conveyed as a target image, but the more detailed consideration and approach are only touched on or assumed (Salvador, 2007). Looking at this now, the multitude of disciplines in mechatronic systems also results in many interfaces that must be regarded. In addition to the interfaces between modules, there are interfaces between components within modules and the components and modules themselves, which require discipline-specific but also cross-discipline exchange (You and Smith, 2016). After defining the modules, development activities can begin in modular product development (Sanchez, 1996). The question arising here is, how to cope with the knowledge and the interaction of various employees over different system and complexity levels.

## 2.3 Collaboration in Engineering

In the development of products, various approaches deal with handling complexity through collaboration or the cooperation of various developers from different departments. To address multiple disciplines involved in the development process, different approaches for complex task solving, e.g., SCRUM, have been developed (Schmidt *et al.*, 2018). Also, DSM literature has included approaches from agile working and evolved into frameworks like Scaled Agile (Smith and Eppinger, 1997; Narayanan *et al.*, 2021). There are more forms of the basic idea of structured collaboration between organizational units and individuals. Approaches in that field recommend the continuous exchange of information and the regular validation of one's own results and the results of the entirety (Narayanan *et al.*, 2021). For example, in SCRUM sprints are used to enforce time-boxing in the execution of tasks and thus provide a continuous opportunity to validate the intended work results and, if necessary, to adapt them to the new requirements and changed findings, even in the design phase (Edin Grimheden, 2013). In addition, in the context of agile working, there is also the principle of distributing responsibility among the teams that are assigned to the tasks. This clear accountability allows decisions to be made more quickly and derived and executed by the direct experts in the design on their own responsibility (Schmidt *et al.*, 2018). However, these decisions should be cascaded and validated in a scalable form and thus also considered and reviewed in the overall context of the entire system (Schmidt *et al.*, 2018; Narayanan *et al.*, 2021).

## 3. Previous Studies and Contribution Focus

This contribution is based on the extended *Design for Variety Framework*, where a system architecture for a robot vacuum cleaner is formed after deciding on relevant product features, defining product functions, developing suitable solutions, and designing components. On the component level of the product architecture, the *Product-DSM* was obtained (Küchenhof *et al.*, 2020). The *Product-DSM* was then extended by mapping the relevant development disciplines in a *Multi-Domain Matrix (MDM)*, including an additional organizational view, as shown in Figure 3, by the dark blue arrows on the left. This *MDM* was transferred into the *Module Harmonization Chart (MHC)*, where components and involved disciplines are mapped, and discipline-specific modules have been formed via harmonization across disciplinary views (Zuefle *et al.*, 2022). The *MHC* is depicted on the right side of Figure 3, which was set in the context of the product life cycle. The paper focuses on extending the existing MDM with further allocations to other domains (for example, responsible employees). This additional allocation allows responsibilities, affiliations, and interactions to be analyzed based on the interfaces and optimized for collaboration. The focus is on product-side and process-side interactions that impact the interfaces in the organizational view addressing collaboration. These interactions are shown in Figure 3 as colored arrows (yellow, pink, orange). Dashed arrows represent the collaboration of an employee concerning a component, and solid arrows represent the collaboration with employees from their discipline.

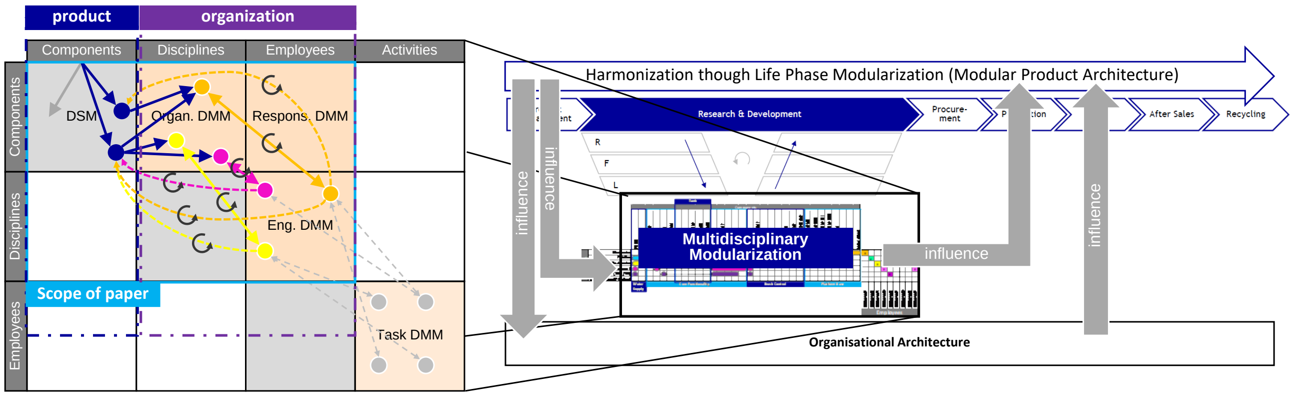

Figure 3. The paper's focus is illustrated by matrix-based fundamentals with product and organization domains (left). Additonal placement in the product life phase, including multi-disciplinary modularization (right).

## 4. Methodical Procedure

For further investigation into the introduced topic, relevant Interfaces and the underlying modular structures can be extracted from the *MHC*, and the development disciplines involved can be allocated. For forming design teams, people or employees are missing in this view. As mentioned in Section 3, the employees/developers involved are mapped to their respective disciplines and related components in a 3x3 MDM. Team constellations can be derived by allocating people to modules and components. The challenge arising is how to synchronize development teams favorably. In the following, a procedure is derived to form team sprints based on components, modules, and their interfaces for supporting responsibilities and activities in interface management.

### 4.1 Derivation of Design Sprints

Therefore, **design sprints** are defined to support the execution and communication of the multi-disciplinary developers involved in design processes, as done by some agile frameworks or similar design techniques, as mentioned in Section 2.4. Because interactions in a modular system architecture take place on different levels and in different scopes, the approach must also be capable of being adapted to these differences. So, a distinction is made between different sprint types depending on the development complexity, e.g., on the level of the system, subsystem, or component level, and temporal organization, e.g., who needs to interact with whom and others. For this reason, the collaboration considered here is broken down into the different system levels: the overall product, the discipline, as a higher-level domain with specific expertise, and the module, as the representation of a defined subsystem. A complete iteration in the V-Model represents a **system sprint** in which all requirements are implemented in the product or the respective prototype and its validation. Based on a modular architecture's scope in this paper, different modules exchange information (*inter-module collaboration*). Due to the capable collaboration in small teams, this sprint idea can also work with a limited number (≤ 9, e.g., limit in SCRUM) of employees in a non-modular architecture. Furthermore, a **discipline sprint** is defined for cross-module incorporating changes and designs in one specific discipline (*inter-module, intra-discipline collaboration*). Here, it can also be possible to use those sprints in non-modular architecture for extended exchange in specific disciplines or domains, e.g., electrical infrastructure. Concluding, the lowest level is the **module sprint** with the highest proportion of capacity. Here the employees from different disciplines meet for frequent exchange on a specific knowledge level within the modules (*intra-module, inter-discipline collaboration).* In this case, a modular architecture could be necessary for an explicit allocation to a subsystem and clear interfaces to other subsystems.

## 4.2 Systematic Assignment to Design Sprints

In the first step, we draw on the harmonized MHC, which has harmonized modules across the disciplines or domains, as a preliminary effort. This preliminary work is a crucial basis since it can ensure that harmonized and uniform module sections are available across all participating disciplines. On the one hand, this is relevant for classifying the employees into modules and preventing discrepancies in the interaction (e.g., different ideas of module sections).

As shown in Figure 4, the interfaces between the modules can be taken from the clustered DSM (center). All the interfaces for module interactions are depicted in the off-diagonals. In this example, components B and C are connected by a yellow interface (e.g., electrical interaction). Therefore, it is clear that this electrical interaction has to be considered wisely due to its effects on the module's overall integration into the system. In addition, between components A and B, there are two interactions, green (e.g., fluidical interaction) and blue (e.g., spatial coupling). Both interfaces are not off-diagonals and have intra-module connections. So, they are not relevant in this first regard. Up to here, this procedure is conventional in matrix-based approaches.

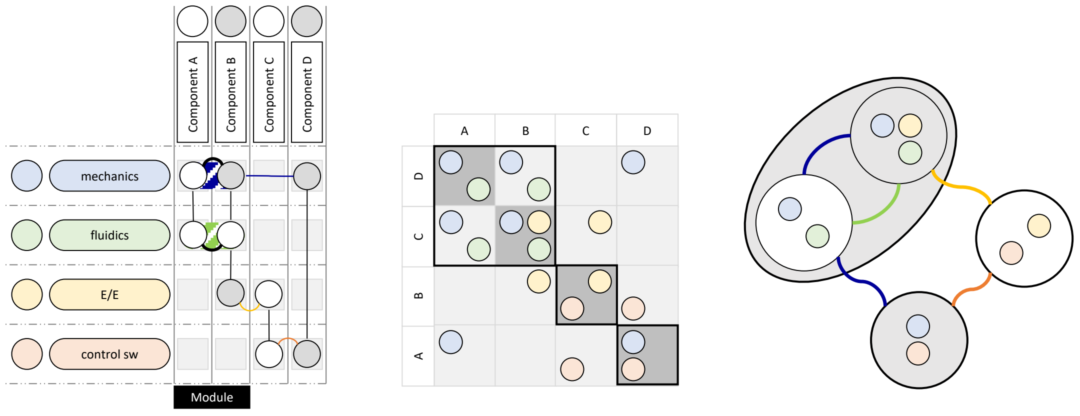

Figure 4. simplified MHC (left), Design-Structure-Matrix (center), and derived system graph with interfaces (right)

In the second step, developers are distributed to the components according to the interfaces. It is claimed that a previously regarded component can have additional discipline-specific components due to the fractional behavior of systems. Here each component was considered due to its discipline-specific components and so its assignment to a specific or a set of specific disciplines, as depicted in Figure 4 (e.g., component A has mechanical and fluidical discipline assignment). That assignment enables allocating the interface and connection between components to specific disciplines and considering special allocations of discipline-specific components to others, resulting in a more transparent view of the system. Afterward, those discipline-specific components can be assigned to specific employees with the necessary expertise.

Now different sprints can be derived. All employees working within a module are assigned to ***module sprints***, which can take place in short and regular intervals, e.g., daily. By reviewing the module interfaces, cross-module communication channels can be made visible. All employees with their share of those cross-module communication activities are divided into ***discipline sprints***. Discipline sprints can take place, e.g., once a week. A ***system sprint*** represents the highest level of communication effort. Here, all developers, project managers, and product owners should be present to evaluate the final concept, which may already be at some prototype level. System sprints, where all modules are expected to work together harmoniously, have the slowest pace and can be held, e.g., once a month or as project needs require, but frequently. Figure 5 shows a generic illustration of this coordination in sprints. On the right side of Figure 5, the different levels of sprints presented in this section are visualized on a simple system graph. It can be seen that sprints cover every level, from clustered modules or single components to discipline-specific interface interaction across different modules and the overall system interaction for efficient integration across modules.

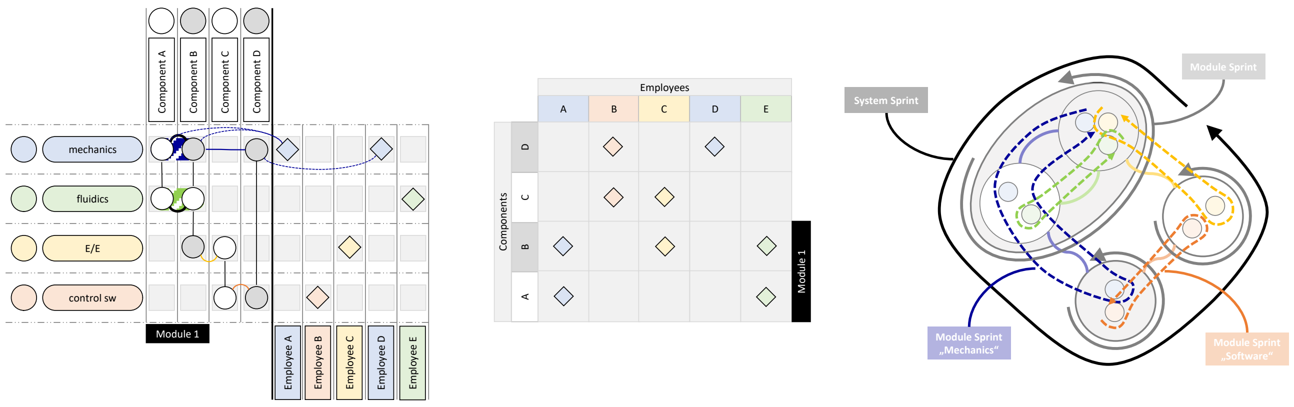

Figure 5. MHC plus employees (left), Domain-Mapping-Matrix (center), and derived system graph with design sprints (right)

## 5. Analysis of Interface Specifications in Exemplary System Architecture

How the procedure described in Section 4 is used on an exemplary system architecture is now demonstrated. Here, it relies on the *MHC* of the robot vacuum cleaner done in previous studies and extends it to include individuals' affiliation to their disciplines and the components they are involved in developing, shown in Figure 7 (Zuefle et al., 2022). The *MHC* in Figure 5 illustrates an abstraction of the 2x2 Multi-Domain-Matrix extended into a 3x3 Multi-Domain-Matrix. This 3x3 matrix allocates the employees working on the exemplary robot vacuum cleaner to the specific discipline and the specific components bound. This allocation is relevant because an employee can work in different disciplines, as it could be in small and medium-sized enterprises (SMEs) or companies working with roles as Systems Engineer, Systems Architect, and Software Architect. Otherwise, assigning a bound of different components or modules to multiple employees of the same discipline is possible. This instance illustrates that the assignment between components or discipline-specific components to employees is the most important allocation due to the goal of clarifying the responsibilities of interfaces. Therefore, creating the 2x2 MDM with components and disciplines, then clustering the modules, and afterward allocating the employees to the modules is necessary.

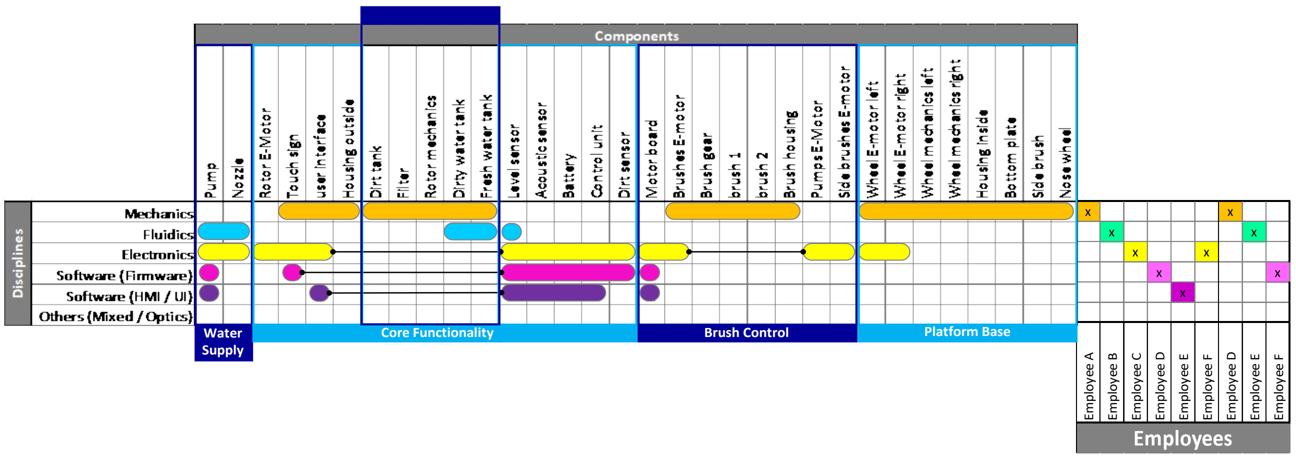

Figure 6. Extended DMM based on 3x3 MDM from Zuefle et al. 2022 by new domain "employees" for assigning employees

Next, the network graph is derived for further investigation. For this, the modules and their interfaces from the used *MHC* (Figure 6) are represented as a system graph (Figure 7). The components from the system architecture are located in the illustrated modules they belong to. The discipline-specific components are marked in their specific color, as in the MHC depicted, and the number of components creating a discipline-specific cluster is highlighted in the system graph. Additionally, the interfaces of the discipline-specific components, as the interfaces between the modules, are visualized and colored in their specific discipline color. For instance, interfaces between components of the same discipline are colored in grey because of their mono-disciplinarity and clear assignment. For increasing transparency and decreasing colored line crossings, only the interfaces between modules are colored. Those interfaces are the off-diagonals in the harmonized multi-disciplinary *DSM*.

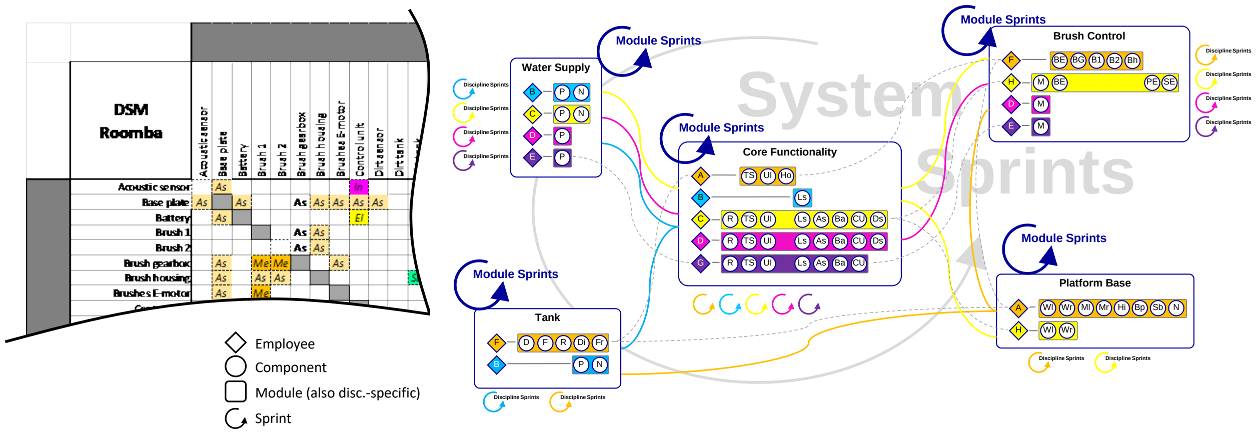

Figure 7. System graph of the different levels of design sprints on the exemplary robot vacuum cleaner

Now the introduced sprint teams can be derived. Due to the allocation of the engineers to the modules/components it is possible to derive if the team cut fits the module cut. For example, looking at the "platform base" module, there are several mechanical components, and it would be counter-productive to have two mechanical engineers responsible. It is way more effective to assign two engineers to two different modules so that the responsibility and expertise in each module are assigned unambiguously. If this is checked, the employees involved in each module are organized into the presented idea of ***module sprints***. In this sprint, the responsibility of intra-module interfaces is clarified. Afterward, the inter-module interfaces in the off-diagonals from the previous DSM can be identified, and all relevant employees affected by them must be assigned to the higher-level ***discipline sprints***. In this sprint, the responsibility of interfaces connecting different modules is clarified. Therefore, the interaction between the discipline-internal employees is relevant for communication about designs, results, and parametrization in their various modules. Furthermore, all the relevant project participants are included in the ***system sprints*** at the highest level in the scope of this system. In addition, it is mentioned that a System here is not necessarily a product itself. In our use case of the vacuum robot, it is a whole product. Thinking about a more complex system like a machine tool, transferring this system level to a subsystem level located as an additional sprint level is recommended. The following definitions illustrate the different Sprints and collaborations using the mentioned robot vacuum cleaner.

## Definition of Module Sprints

Module sprints aim to be able to develop the module itself as a functioning subsystem. Regarding the black box approach, the focus is exclusively on the module with its relevant requirements and functions. All disciplines in the respective module exchange information on a short cycle level to achieve the best possible cooperation in the system context.

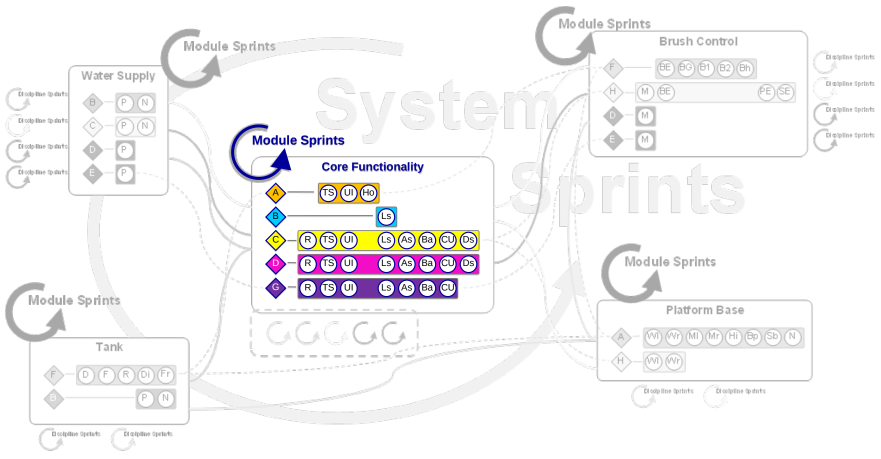

Figure 8. Module core functionalities as an example of the constellation of one module sprint

Due to the different disciplines involved, going into external interfaces here is not expedient. The reason for this is that the responsibility of the interface must be assigned according to the interface type. For example, a software developer should not be responsible for spatial coupling. Only internal interfaces are discussed in a Module sprint, affecting mutual interactions through design decisions. Figure 8 shows an example module sprint in the "Core Functionalities" module. As seen here, no interfaces to other modules are part of this sprint. But the internal interfaces must be considered in this sprint for efficient development of the module functionality.

## Definition of Discipline Sprints

Discipline sprints aim to work out how to ensure effective interactions between modules by discussing and developing working interfaces between multiple modules. In this process, the interface's responsibility lies with the staff member or the module itself, which creates the need for the interface. For example, Figure 10 shows that the Core Functionality module requires the Brush Control module to implement the brushed floor cleaning functionality. The responsibility for this interface and its coordination lies with the first module. The responsibility at the employee level then lies with the employee for the specific discipline of the interface and the triggered module. In this figure, it is employee "C". As can also be seen from Figure 9, it is possible to have an interface between two modules with the same discipline-specific employee as the person responsible. So, the responsibility potentially does not need to be defined, but if the assignment changes, it still needs to be clarified.

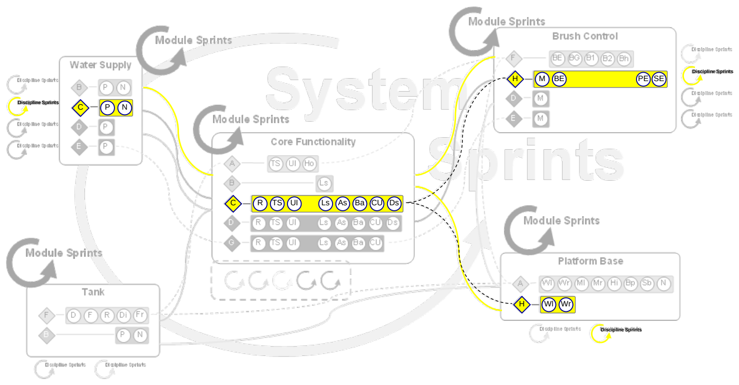

Figure 9. Discipline sprint in electrical submodules across four modules.

## Definition of System Sprints

Compared to the other two design sprint levels, the system sprint looks at the effects of all modules on the overall system plus several subsystems in the case of more extensive systems.

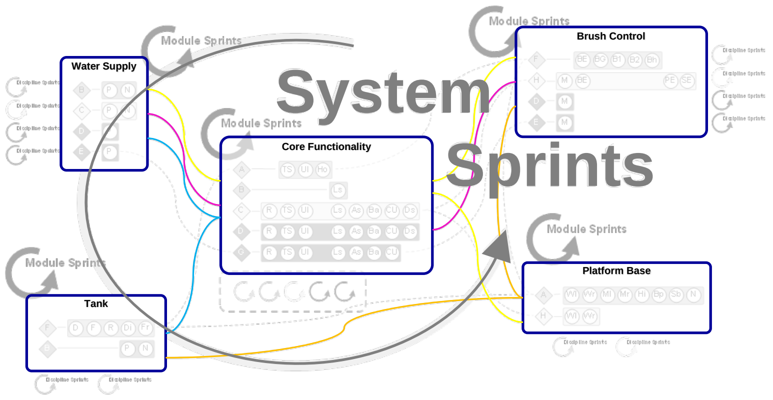

Figure 10. The system sprint as integration and coordination of all five modules

The focus is on integrating the modules into the overall working system, including verifying and validating the previously done development. The system sprint elaborates that the system can provide its functionality. This goal includes the communication about the interaction of modules, the communication of the interaction of disciplines across modules (from discipline sprints), and the verification and validation of implemented and defined interfaces. Figure 10 depicts this design sprint based on the exemplary product family of the robot vacuum cleaner. It can be seen that the holistic interaction of the modules is the targeted focus and not several interactions between discipline-specific module scopes and intra-module interactions. This clear objective focuses on relevant information at each design sprint level and helps rely on essential actions.

## Additional Time Harmonization

Both module and discipline sprints can be subject to different temporal intensities. For example, development cycles in software development are generally run through more frequently and faster than in mechanical design. Depending on the development effort (e.g., programming or design) and discipline-related complexity, appropriate periods should be provided. This results in the possibility of decoupling discipline-specific sprints and allows more flexibility (e.g., software engineers can meet twice a week, mechanical engineers only once a week) in organizing different topics but keeping everything within a holistic framework.

## 6. Discussion & Outlook

Harmonizing development activities, especially with different development disciplines from other knowledge domains, represents a significant challenge in product design. Primarily through new technologies and rising digital shares, the rising systems complexity must be considered by structuring products accordingly and integrating the people collaborating for successful product development. This contribution brings elements from the development of modular product families, mechatronic systems, and the organizational view to cope with the systems' complexity and harmonize development activities on the bottom of the V-model. A procedure has been shown to map people to their knowledge field/ development discipline and relate them to the system under development. Then sprint teams based on the module, component interfaces, and allocated developers have been formed. The method was applied to a vacuum cleaner robot as an exemplary system architecture and derived a sprint plan. The interface and component specification can be supplemented by more development data such as effort, e.g., working hours to complete a component, module, and their interfaces. This approach can also be done with the help of model-based systems engineering. All relevant elements shown here can be implemented and linked in a systems model in SysML. Development data can then be stored consistently and centrally. Also, the created models can be easily extended, and metadata for the elements can be given to build more complex system models for parametrized work scheduling to develop complex systems such as CPPF. The relations in the MDM between the product and organization domain represent a flow of information that can be very valuable for systems design. A central question is how the information needs to be distributed. Both physical and virtual assets in a complex product need structures and processes to function correctly. Also, both need to be designed by multiple people; we see their coordination and communication as critical success factors for developing future products and systems.

## References

Askhøj, C., Christensen, C.K.F. and Mortensen, N.H. (2021), “Cross domain modularization tool: Mechanics, electronics, and software”, *Concurrent Engineering*, 1063293X2110003.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J. and Thomas, D. (2001), *Manifesto for Agile Software Development*, available at: http://www.agilemanifesto.org/.

Browning, T.R. and Yassine, A.A. (2016), “Managing a Portfolio of Product Development Projects under Resource Constraints”, *Decision Sciences*, Vol. 47 No. 2, pp. 333–372.

Cabrera, A.A.A., Erden, M.S., Foeken, M.J. and Tomiyama, T. (2008), “High Level Model Integration for Design of Mechatronic Systems”, in *2008 IEEE/ASME International Conference on Mechtronic and Embedded Systems and Applications, 12.10.2008 - 15.10.2008, Beijing, China*, IEEE, pp. 387–392.

Cadavid, H., Andrikopoulos, V., Avgeriou, P. and Broekema, P.C. (2021), “System- and Software-level Architecting Harmonization Practices for Systems-of-Systems An exploratory case study on a long-running large-scale scientific instrument”, in *2021 IEEE 18th International Conference on Software Architecture (ICSA), 22.03.2021 - 26.03.2021, Stuttgart, Germany*, IEEE, pp. 13–24.

Danilovic, M. and Browning, T.R. (2007), “Managing complex product development projects with design structure matrices and domain mapping matrices”, *International Journal of Project Management*, Vol. 25 No. 3, pp. 300–314.

Edin Grimheden, M. (2013), “Can agile methods enhance mechatronics design education?”, *Mechatronics*, Vol. 23 No. 8, pp. 967–973.

Eigner, M., Dickopf, T., Apostolov, H., Schaefer, P., Faißt, K.-G. and Keßler, A. (2014), “System Lifecycle Management: Initial Approach for a Sustainable Product Development Process Based on Methods of Model Based Systems Engineering”, in Fukuda, S., Bernard, A., Gurumoorthy, B. and Bouras, A. (Eds.), *Product Lifecycle Management for a Global Market*, *IFIP Advances in Information and Communication Technology*, Vol. 442, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 287–300.

ElMaraghy, W., ElMaraghy, H., Tomiyama, T. and Monostori, L. (2012), “Complexity in engineering design and manufacturing”, *CIRP Annals*, Vol. 61 No. 2, pp. 793–814.

Eppinger, S.D., Joglekar, N.R., Olechowski, A. and Teo, T. (2014), “Improving the systems engineering process with multilevel analysis of interactions”, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol. 28 No. 4, pp. 323–337.

Erixon, G. (1998), *Modular function deployment: A method for product modularisation*, Zugl.: Stockholm, Kungl. Tekn. Högsk., Diss., 1998, *TRITA-MSM*, Vol. 98,1, The Royal Inst. of Technology Dept. of Manufacturing Systems Assembly Systems Division, Stockholm.

Graessler, I., Hentze, J. and Bruckmann, T. (2018), “V-MODELS FOR INTERDISCIPLINARY SYSTEMS ENGINEERING”, in *Proceedings of the DESIGN 2018 15th International Design Conference, May, 21-24, 2018*, Faculty of Mechanical Engineering and Naval Architecture, University of Zagreb, Croatia; The Design Society, Glasgow, UK, pp. 747–756.

Hehenberger, P., Vogel-Heuser, B., Bradley, D., Eynard, B., Tomiyama, T. and Achiche, S. (2016), "Design, modelling, simulation and integration of cyber physical systems: Methods and applications", *Computers in Industry*, Vol. 82, pp. 273–289.

Helmer, R., Yassine, A. and Meier, C. (2010), "Systematic module and interface definition using component design structure matrix", *Journal of Engineering Design*, Vol. 21 No. 6, pp. 647–675.

Krause, D., Beckmann, G., Eilmus, S., Gebhardt, N., Jonas, H. and Rettberg, R. (2014), "Integrated Development of Modular Product Families: A Methods Toolkit", in Simpson, T.W., Jiao, J., Siddique, Z. and Hölttä-Otto, K. (Eds.), *Advances in Product Family and Product Platform Design: Methods & applications*, *Advances in Product Family and Product Platform Design*, Springer New York, New York, NY, pp. 245–269.

Krause, D. and Gebhardt, N. (2023), *Methodical Development of Modular Product Families*, Springer Berlin Heidelberg, Berlin, Heidelberg.

Küchenhof, J., Berschik, M.C., Heyden, E. and Krause, D. (2022), "METHODICAL SUPPORT FOR THE NEW DEVELOPMENT OF CYBERPHYSICAL PRODUCT FAMILIES", *17TH INTERNATIONAL DESIGN CONFERENCE*.

Küchenhof, J., Tabel, C. and Krause, D. (2020), "Assessing the Influence of Generational Variety on Product Family Structures", *Procedia CIRP*, Vol. 91, pp. 796–801.

Lambe, A.B. and Martins, J.R.R.A. (2012), "Extensions to the design structure matrix for the description of multi-disciplinary design, analysis, and optimization processes", *Structural and Multi-disciplinary Optimization*, Vol. 46 No. 2, pp. 273–284.

Mertens, K.G., Rennpferdt, C., Greve, E., Krause, D. and Meyer, M. (2022), "Reviewing the intellectual structure of product modularization: Toward a common view and future research agenda", *Journal of Product Innovation Management*.

Mikkola, J.H. and Gassmann, O. (2003), "Managing modularity of product architectures: toward an integrated theory", *IEEE Transactions on Engineering Management*, Vol. 50 No. 2, pp. 204–218.

Narayanan, N., Joglekar, N. and Eppinger, S. (2021), "Improving Scaled Agile with Multi-Domain Matrix", in *DS 112: Proceedings of the 23rd International DSM Conference (DSM 2021), Montréal, Canada, October, 12 - 14, 2021, October 12 - 14 2021*, The Design Society.

Otto, K., Hölttä-Otto, K., Simpson, T.W., Krause, D., Ripperda, S. and Ki Moon, S. (2016), "Global Views on Modular Design Research: Linking Alternative Methods to Support Modular Product Family Concept Development", *Journal of Mechanical Design*, Vol. 138 No. 7.

Salvador, F. (2007), "Toward a Product System Modularity Construct: Literature Review and Reconceptualization", *IEEE Transactions on Engineering Management*, Vol. 54 No. 2, pp. 219–240.

Schmidt, T.S., Weiss, S. and Paetzold, K. (2018), *Agile Development of Physical Products: An Empirical Study about Motivations, Potentials and Applicability*, Universitätsbibliothek der Universität der Bundeswehr München, Neubiberg.

Simpson, T.W., Jiao, J., Siddique, Z. and Hölttä-Otto, K. (Eds.) (2014), *Advances in Product Family and Product Platform Design: Methods & applications*, *Advances in Product Family and Product Platform Design*, Springer New York, New York, NY.

Smith, R.P. and Eppinger, S.D. (1997), "Identifying Controlling Features of Engineering Design Iteration", *Management Science*, Vol. 43 No. 3, pp. 276–293.

Sosa, M.E., Eppinger, S.D. and Rowles, C.M. (2003), "Identifying Modular and Integrative Systems and Their Impact on Design Team Interactions", *Journal of Mechanical Design*, Vol. 125 No. 2, pp. 240–252.

Sosa, M.E., Eppinger, S.D. and Rowles, C.M. (2004), "The Misalignment of Product Architecture and Organizational Structure in Complex Product Development", *Management Science*, Vol. 50 No. 12, pp. 1674–1689.

Tomiyama, T., Lutters, E., Stark, R. and Abramovici, M. (2019), "Development capabilities for smart products", *CIRP Annals*, Vol. 68 No. 2, pp. 727–750.

van Beek, T.J., Erden, M.S. and Tomiyama, T. (2010), "Modular design of mechatronic systems with function modeling", *Mechatronics*, Vol. 20 No. 8, pp. 850–863.

Yassine, A.A. (2021), "Managing the Development of Complex Product Systems: An Integrative Literature Review", *IEEE Transactions on Engineering Management*, Vol. 68 No. 6, pp. 1619–1636.

You, Z.-H. and Smith, S. (2016), "A multi-objective modular design method for creating highly distinct independent modules", *Research in Engineering Design*, Vol. 27 No. 2, pp. 179–191.

Zuefle, M. and Krause, D. (2023), "Multi-Disciplinary Product Design and Modularization – Concept Introduction of the Module Harmonization Chart (MHC)", *Procedia CIRP*, Vol. 119, pp. 938–943.

Zuefle, M., Küchenhof, J., Hanna, M. and Krause, K. (2022), "Assessing the Influence of Digital Innovations on the Organizational Design of Product Family Generations", in *DS 121: Proceedings of the 24th International DSM Conference (DSM 2022), Eindhoven, The Netherlands, October, 11 - 13, 2022, October 11 - 13 2022*, The Design Society, pp. 58–67.

**Contact: M. Zuefle,** Hamburg University of Technology (TUHH), Institute of Product Development and Mechanical Engineering Design (PKT), Denickestraße 17, 21071 Hamburg, Germany, +49-40-42878-3231, +49-40-42878-2296, marc.zuefle@tuhh.de, www.tuhh.de/pkt