NordDesign 2020
August 11-14, 2020
Kgs. Lyngby, Denmark

Planning & Tracking the Changes - Matrix Mapping of Modular Product Family Generations

Jan Küchenhof, Lea-Nadine Schwede, Dieter Krause

PKT - Institute for Product Development and Mechanical Engineering Design,
Hamburg University of Technology (TUHH), Germany
{jan.kuechenhof; lea.schwede; krause}@tuhh.de

Abstract

In this paper, the handling of different product family generations is methodically supported. In order to prepare low internal complexity for future product variants, Design for Variety and modularisation approaches are used to build a multiple domain matrix framework to structure a product family initially. By introducing Change- and Delta-Matrices, the planning as well as the tracing of changes between two different product family generations can be displayed. For supporting the development process of complex systems, the approach of Model-Based Systems Engineering is used. The activities are assisted by means of graph analysis software to obtain change propagation paths and SysML to ensure data consistency and track changes respectively.

Keywords: Product Families, New Product Development, Model-Based Systems Engineering, Design Structure Matrix (DSM)

1 Introduction

Shorter product life cycles, emerging technologies as well as user needs and customer requirements require companies to change their products, processes and even their organizational structure more and more frequently. Introducing new product generations is a possible answer to satisfying the new resulting requirements. This early phase of product development is accompanied by high uncertainty as the degree of freedom is high but the result of action is unknown as the system behaviour is not well known. Ehrlenspiel and Meerkamm explain this design paradoxon as the product properties can be influenced most strongly in the first phases of the product life cycle, i.e. in planning and design, and the effort required to change them is still the lowest there, but the possibility of gaining knowledge about the product properties to be defined is generally lowest precisely in these phases (Ehrlenspiel and Meerkamm, 2013). To satisfy a high external product variety with little internal product and process variety, modular product structuring with help of Design for Variety (DfV) pose a possible solution (Krause and Gebhardt, 2018). As the product properties are part of an early design domain during development and their definition is crucial for downstream design activities, the effect of changes in the domain of offered product features on the overall defined systems is of high interest. As product components pose a special role for product development, this domain needs particular consideration. Necessary changes within the system may be small

or large but estimating their effect can be challenging, especially when designing complex products and modular product families are seen as such. The traceability of these changes and their consequences is a current research problem. In this paper, the effects of changes for two developed product family generations (PFGs) are investigated based on a real development case with help of Model-Based Systems Engineering (MBSE) using the modelling software Cameo Systems Modeler (CSM) and the graph analysis and visualisation software Cytoscape . Section 2 describes the state of the art in the design of complex systems and modular product development. The basics of methods for matrix structuring and suitable software support are presented. In Section 3, the methodical extension for planning and tracking changes within PFG development with help of matrix-based structuring is shown and applied on a use-case. Cytoscape is used to display change propagation paths in a network graph of the multiple domain matrices and visualise the effects of new objectives and requirements on the product structure to support planning activities within the product creation process. CSM is used for data structuring and storage, exchange and to trace the changes after development activities. Here, the differences within the two developed product structures are displayed with support of the software dependency handling and visualisation in matrices. This paper concludes with a discussion and outlook in Section 4.

2 State of the Art

The following section is divided into four subparts. Starting with methodical support for the development of modular products, the basis for product generation engineering and corresponding mechanisms of change are presented. Matrix-based structuring methods and the use of appropriate software are furthermore explained. The section closes with the presentation of the preparatory studies leading to the here depicted development case.

2.1 Modular Product Design and Product Generation Engineering

The development of modular product families can be supported by various methods such as the technical-functional modularisation based on function heuristics according to Stone (Stone, 1997) or the product-strategic Modular Function Deployment after Erixon (Erixon, 1998). The Integrated PKT-Approach for Development of Modular Product Families combines technical-functional and product-strategic aspects of modularisation. The methodical approach can be subdivided into two main parts: Variance-oriented component structuring with help of DfV and the subsequent Life Phase Modularisation, taking into account different perspectives of the product life phases and harmonizing the overall modular structure (Krause and Gebhardt, 2018). DfV after Kipp is briefly explained as it is basis for the further procedure in this paper. DfV aims to reduce the internal component variety of product families while retaining the external sales offer. In order to align the product structure with the external variety of offers, it is first visualised in the form of a Tree of external Variety (TeV). In addition, the variance of functions is recorded by adding the variance of the individual functions to a flow-based functional structure (PFFS). For this purpose, the functions are divided into standard and variant functions. At component level, the variance of the components of the product family is recorded and displayed in the form of a Module Interface Graph (MIG). The variety of the components of the product family on each design level is transferred into the four level Variety Allocation Model (VAM) and the correlation between external and internal variety can be analysed (Kipp. 2012). The re-design effort rises with reaching higher levels of the VAM. While changes in components and principle solutions (level 4 and 3) occur most frequently, changes in functions and features (level 2 and 1) are related to higher expenditures and point towards new development activities (Kipp, 2012). Modular product structuring is usually carried out on grown product families. In order to support the new development of modular product families,

where architectural knowledge about components and their interactions is seen as input to development activities and not output of them (Sanchez and Mahoney, 1996), the product architecture needs to be set up before the development starts. To do that, DfV after Kipp (Kipp, 2012) has been extended towards the initial structuring of modular product families (Küchenhof and Krause, 2019). In new development activities, models for each level need to be initially designed and start with the first level of product features, following the subsequent design stages down to the component level creating an initial product structure. Although Kipp omits standard parts in the VAM, they are integrated within the new development as component and interface variety is not defined and needs to be assessed. A further differentiation into variant and specific design parts is made in order to be able to offer customised designs (Küchenhof and Krause, 2019). A well-defined product structure may be a good starting point but early product generations often lack maturity, successive product generations need to be developed in order to be truly successful (Albers et al., 2016). A new product generation is thus consisting of subsystems that are the result of variation in order to *carryover*, subsystems that are new developed with *embodiment variation* or *principle variation* (Albers et al. 2016). The scope of a development project can be assessed leaning on Pahl and Beitz‘s characterisation into the design types *new design*, *adaptive design*, *variant design* and *repeated design* (Feldhusen and Grote, 2013). As a whole product family is under consideration and not just a single product, the scope of and need for action are different. Change mechanisms with respect to product families can be found in Du et al. (2001). Here, component and module variety generations are described by the three basic mechanisms *attaching/ removing*, *swapping* and *scaling*. To *attach* or *remove* modules to a base product enables or disables certain functions; the interfaces must therefore be designed appropriately. Modules or components that fulfil different performance requirements but carry out the same function can be substituted or swapped with corresponding counterparts. *Swapping* can be seen as a combination of attaching and removing; common interfaces are required for interchangeability. Changing a module or a product to certain operational parameters can be described as *scaling*. A more complicated mechanism, composed by employing the first mechanisms recursively is referred to as *variety nesting*. Here, multiple modules and different hierarchical levels are encompassed as components within modules are affected (Du et al., 2001).

2.2 Matrix-Based Product Structuring

Since the product, or furthermore the product family, is characterised by a high number of interacting elements, and complex products show a dynamic change in the parameters of the interactions, the object under consideration can be understood as a complex system (Lindemann et al., 2009). An approach to deal with product complexity is the Structural Complexity Management after Lindemann et al. (2009). A considerable part of complexity results from dependencies between system elements, since the adaptation of single system elements can cause far-reaching consequences within the system. Those are often hardly considered if they are not visually analysed (Lindemann et al., 2009). A popular method to represent and analyse complex systems is the Design Structure Matrix (DSM), which has been applied in various application fields as can be comprehended in Browning (2016). A DSM is a square matrix representing system elements in the diagonal cells such as product components, processes or organisational structures and the dependencies on the off-diagonal entries. Inter- and intra-domain relationships can be analysed in multidomain matrices (MDMs) (Browning, 2016). Since the focus in this paper is on changes, looking further into the DSM literature leads to the ΔDSM and Change-DSM presented in de Weck (2007). The ΔDSM displays the difference between a baseline system and a changed system. With help of system graphs, Change-DSMs can be used to show change propagation paths. Here, the initiating components are displayed

in the columns and the receiving components are displayed as rows in the matrix and thus a direction is assigned to the dependencies (de Weck, 2007). Orawski et al. (2012) take up this concept and apply it with MDMs for life-cycle oriented requirements formalisation and show how traceability of the changes as well as the impact of future changes on the product impacts characteristics of a product (Orawski et al., 2012). This is the base for the later applied MDM architecture on the development case. The need for management of the accruing data is explained in the next section.

2.3 Model-Based Systems Engineering: Data Consistency

For supporting the development process of complex systems, the approach of MBSE can be used. A system in this case consists of a system architecture, requirements and a system behaviour. The system modelling language (SysML) was developed in MBSE to describe and analyse complicated system relationships. SysML is used to define system elements and the links between them (Weilkiens, 2008). SysML models can be designed using the CSM, for example (Holt, 2012). The definitions are made in different diagrams, such as block definition diagrams. In addition, the CSM also allows data relationships to be displayed in matrices. The software support enables the traceability of system elements and consistency, which is assured in different dimensions: Consistency of time, vertical consistency and consistency between multiple models. With the consistency of time, an extension of the product family is possible. Vertical consistency guarantees that the partial models are consistent across different hierarchical levels. Similarly, consistency between multiple models is enabled by the fact that each model element exists only once, which is also known as the ‘single source of truth’ (Bursac 2016, Scherer 2016). MBSE approaches have already been used in product generation development. Bursac modelled different product generations within his scope of variation principles based on existing structures in SysML (Bursac, 2016). The generations depicted are of course different from each other. However, the differences were not pursued further or examined with regard to the objectives to be achieved.

2.4 New Development and Software Support

The initial conceptualization of an early phase product structure with help DfV tools and software support with SysML is shown in (Küchenhof et al., 2019). The modular product structure is obtained, carrying out the relevant design steps of DfV initially and by applying the technical-functional heuristics after Stone (1997) with regard to product variety on the PFFS. The accruing data is stored in CMS and the component dependencies are visualised as a DSM (Küchenhof et al., 2019). Based on the resulting first PFG MorTy as the reference system, a second PFG, the Design Education Platform (DEP), was developed for an interdisciplinary design course in order to enable future product developers to understand design theory and methodology as well as the development of complex systems and manufacturing with 3D-printing to offer high design flexibility (Heyden et al., 2020). The methodical support of the design iteration of the two PFGs with help of MDMs and appropriate software are shown in the following.

3 Methodical Extension & Application on Use-Case

In order to support the new product family development, the consciousness about changes is seen as relevant regarding the planning of PFGs as well as the learning from experience. The transfer of the development tools of the DfV into matrix-based tools was presented in Küchenhof et al. (2019) using the planned modular robot family MoRty and can be comprehended on the left in Figure 1. The MDM development framework in the middle is

leaned on the development case presented in Küchenhof et al. (2020), considering multiple design domains and external as well as internal variety. The concept of Change- and Delta-matrices presented in 2.2 is merged with the already developed building blocks. This expansion can be seen on the right in Figure 1. The relationship between the four MDM blocks on the right is explained in the following and then demonstrated in the application case.

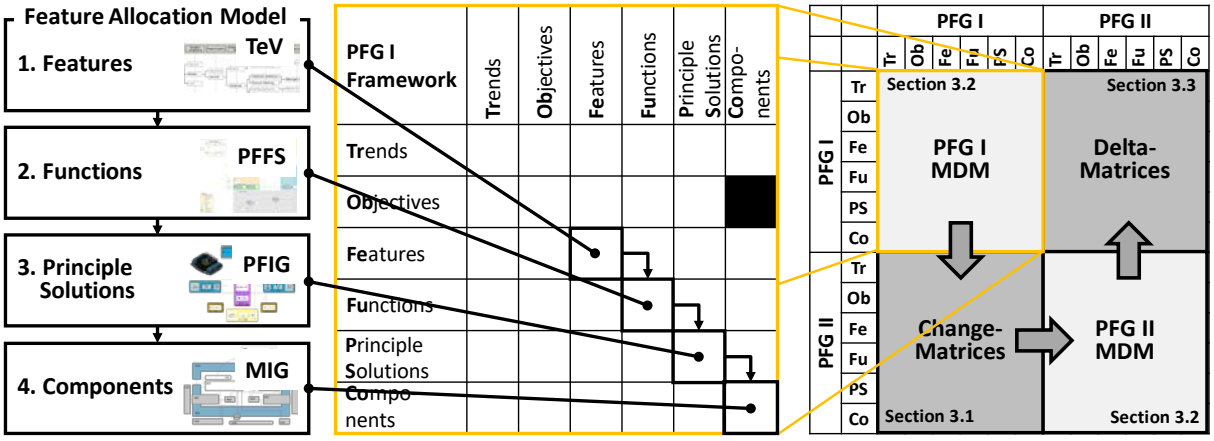

Figure 1: Left: Design domains of the Feature Allocation Model and corresponding development tools. Middle: Matrix framework inhibiting the design domains of DfV complemented by Trends and Objectives. Right: MDM framework of Product Family Generations with Change- and Delta-Matrices

PFG I MDM The matrix segment in the top left represents the MDMs of the first PFG. The considered domains are trends and objectives as well as the subsequent design levels of DfV: features, functions, principle solutions and components. Matrix arrays where the same domains meet represent DSMs each with the product component DSM in the lower right corner. The linkages between two domains can be comprehended by examining the corresponding array, e.g. features-functions.

Change-Matrix (feedforward) On the bottom left, the Change-Matrix is located. Here, planned development activities can be opposed to a baseline system. The first PFG serves as such. Changes occur on all levels, and can be traced on each level accordingly. We are interested in understanding the impact of objectives and new requirements onto product components and the changes in the product component domain in particular, as this level is usually focussed in product development. The influence is further evaluated with help of change propagation paths represented in Cytoscape.

PFG II MDM The matrix on the bottom right represents the MDMs for the second PFG. It inhibits the same domains and the same granularity of observation but is filled with the dataset from the development activities for the DEP.

Delta-Matrix (feedback) The Delta-Matrix is positioned in the top right segment of the MDM. Only after development activities, the full data set can be implemented in CSM and the differences between target and actual system can be traced here and will be shown later comparing the reference system MoRty and the changed system of the DEP.

3.1 New Requirements – Propagation of Change

The data set from the first PFG framework presented in Figure 1 can be taken from CMS, exported to Excel sheets and then imported into the graph analysis and visualisation software Cytoscape since the software offers appropriate interfaces. Figure 2 shows an excerpt of the PFG I MDM, which was visualised in Cytoscape. Although trends were mentioned earlier and these have a great influence on the development case, especially regarding the development of cyber-physical, networked systems and in the field of 3D-printing, these are not linked in the network. The elements of the considered areas are each assigned a different shape, which can

be determined using the legend. The domains are linked corresponding to the deposited matrices. The different colouring indicates the variance of each element and varies from standard elements, which are used only once in the PFG, to variant elements, which contain a predefined set of elements from which can be chosen, further to specific elements, which occur only once in a PFG. For further explanation, the change propagation path for one development objective is shown in Figure 2 and can be comprehended with help of the network graph.

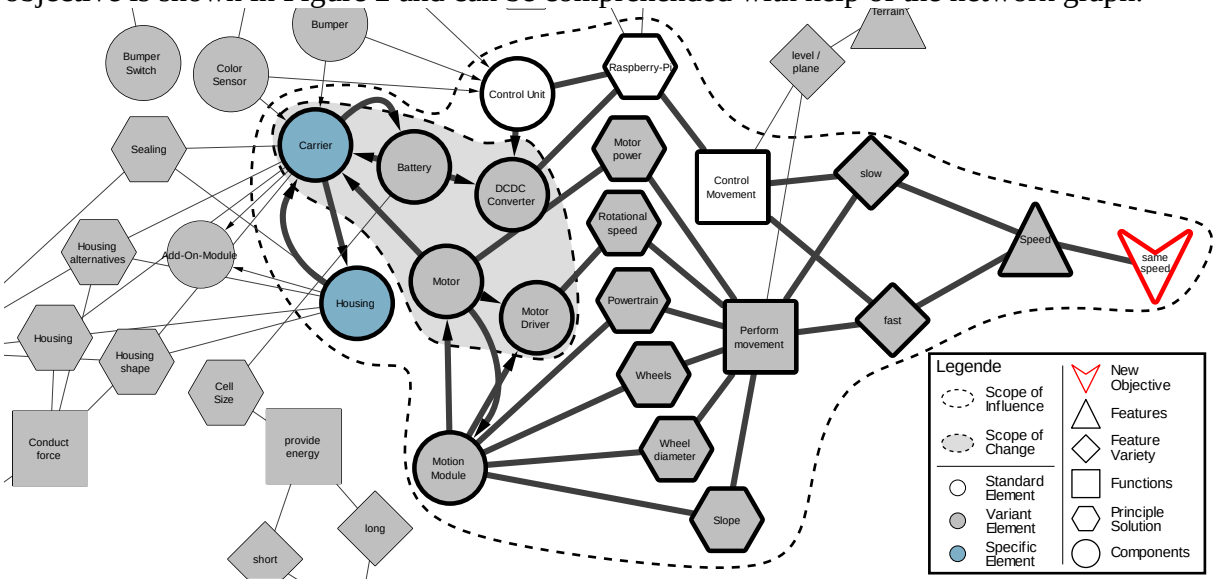

Figure 2: Change Propagation Paths visualised in Cytoscape

The development of the DEP takes place with different boundary conditions and objectives than the initial product family development and there are many requirements attached to the development case. The final event is a parkour challenge for the student teams at the end of the design course. The initial conditions for the student teams should be the same, hence, one development objective can be defined as *same conditions regarding movement speed of the units.* The new objective is integrated in the network (marked red in Figure 2) and is linked to the product feature *speed* from the first PFG. Via the connected functions and principle solutions, the components influenced by the new objective setting can be traced in the visualised network graph. The new objective *same initial speed conditions* links to the (in the first PFG predefined) product feature *speed* and links to the product feature characteristics *slow* and *fast*, which relate to the functions *control movement* and *perform movement*. The solution for the first function in the first PFG is a *RaspberryPi*, which is a standard element in that domain (indicated by the white colour in the figures) for all robot units and defined as the platform module, since the *control unit* is basis for the standardised wireless open and closed loop control, which was a main requirement for that development case. The second influenced function *perform movement* is a variant function and leads to several variant solution principles and the subsequent variant and specific components lying in the change propagation path encircled as *scope of influence* in Figure 2 Since the goal for standardisation would conflict with varying the control function, the two options following the variant path mainly lead to components connected with the *motor* and the *motion module*. Latter contributes to the movement properties as *wheel diameter* and the motion principle, providing the options of *wheels* and a combined solution with a *powertrain* as chosen in the first PFG, can be varied. As part of the design course is the 3D-modelling and -printing of the *housing* and the students are encouraged to be creative about the design, this component is decided to be left a specific component. In order to ease the internal procurement and technical handling of the robot units the *motor* is chosen to be standardised. Following the effect structure in Figure 2, other related components can be identified, lying in the change propagation path, which is marked as *scope*

of change in the figure. Since the *motor* is now a standard component, the *motor driver* can be standardised as well. The same choice was made for the *battery* and the connected *DCDC-converter*, which needs to be adapted to the chosen *battery*. To protect the sensible electrical components from external disturbances, including the students who mostly stem from mechanical engineering design classes, the *carrier* is also standardised and shall contain the components lying in the scope of change plus the *control unit*. The motion module is left as variant for the students to find the best solution to get the most speed out. The design decision is now transferred to the MIG, shown in Figure 3 and serves as the design template for the students. The standard parts are not to touch for the students and those components marked variant leave parametrised scaling options. The blue housing has the highest freedom of design as it can be completely 3D-printed and is marked specific accordingly. In order to make the platform concept work, the interfaces between the components also need to be adapted. The derived requirement from the standardisation for the design parts left for the students is, that the interfaces from the connected components fit the subsequently standardised platform interfaces. After this planning phase, the design and manufacturing can take place which has been successfully carried out for the first time in terms of the interdisciplinary teaching concept as can be followed in Heyden et al. (2020). To be able to track the performed changes comprehensively, the new product generation framework from the DEP is also implemented in CMS. Changes on all design domains can now be traced, stored and made accessible for knowledge management for the development of further PFGs.

3.2 Visual Support of the Interim Development Activities with Help of the MIG

After the defining the scope of change with help of the graph-based change propagation, a MIG for the second PFG is drawn as design and explanatory document as this tool is especially developed for interdisciplinary communication and therefore suited for the design case. The MIGs from the first PFG MoRty as well as for the second PFG DEP can be seen in Figure 3.

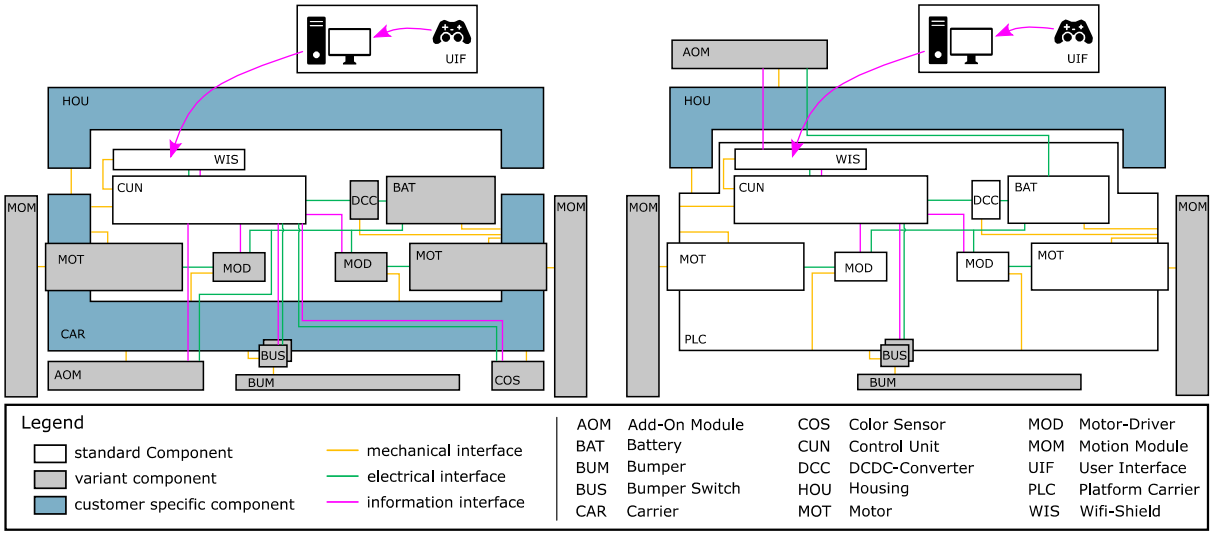

Figure 3: Product Structure of the two product family generations MoRty (left) and the DEP (right) each represented in the Module Interface Graph (MIG)

The product structure is shown in terms of the components and their mechanical, electrical and information interfaces as well as their spatial relation and reference to product family variety according to the definition given in 3.1. The MIG is designed to support the communication of complex content in an understandable way. However, the complexity of real development cases is higher and therefore requires a higher level of support provided by software. CSM is used as the database for successive PFGs. The new linkages can easily be exported into excel sheets

and made available for change propagation with the change-matrix for a new development activity supported by the network graph as described in 3.1 at any given time. In the following, the tracking of conducted changes with focus on product component variety and interface displacement in the product structure with help of SysML is shown.

3.3 Development Completed? - Tracking the Changes

Figure 4 shows the data relationships in the component domains in CMS using a dependency matrix. The matrix shows the DSM for PFG I (upper left), the DSM for the PFG II (bottom right) and the Delta-Matrix for the component domain (upper right). With help of the legend, the matrix entries can be assigned to properties that are unique in the entire SysML-model. The matrix entries within the PFGs represent the structural interfaces, which can also be seen in the MIGs in Figure 3. The same colours as in the MIGs were chosen for the different types of interfaces. The numbers at the edge of the respective DSMs show how many links have been set in the matrix. The DSMs are symmetrical matrices, since no direction is specified for the interfaces. For example, the *add-on module* from MoRty has three interfaces: electrical to b*attery*, mechanical to *carrier* and information to the *control unit* (Figure 4, ①).

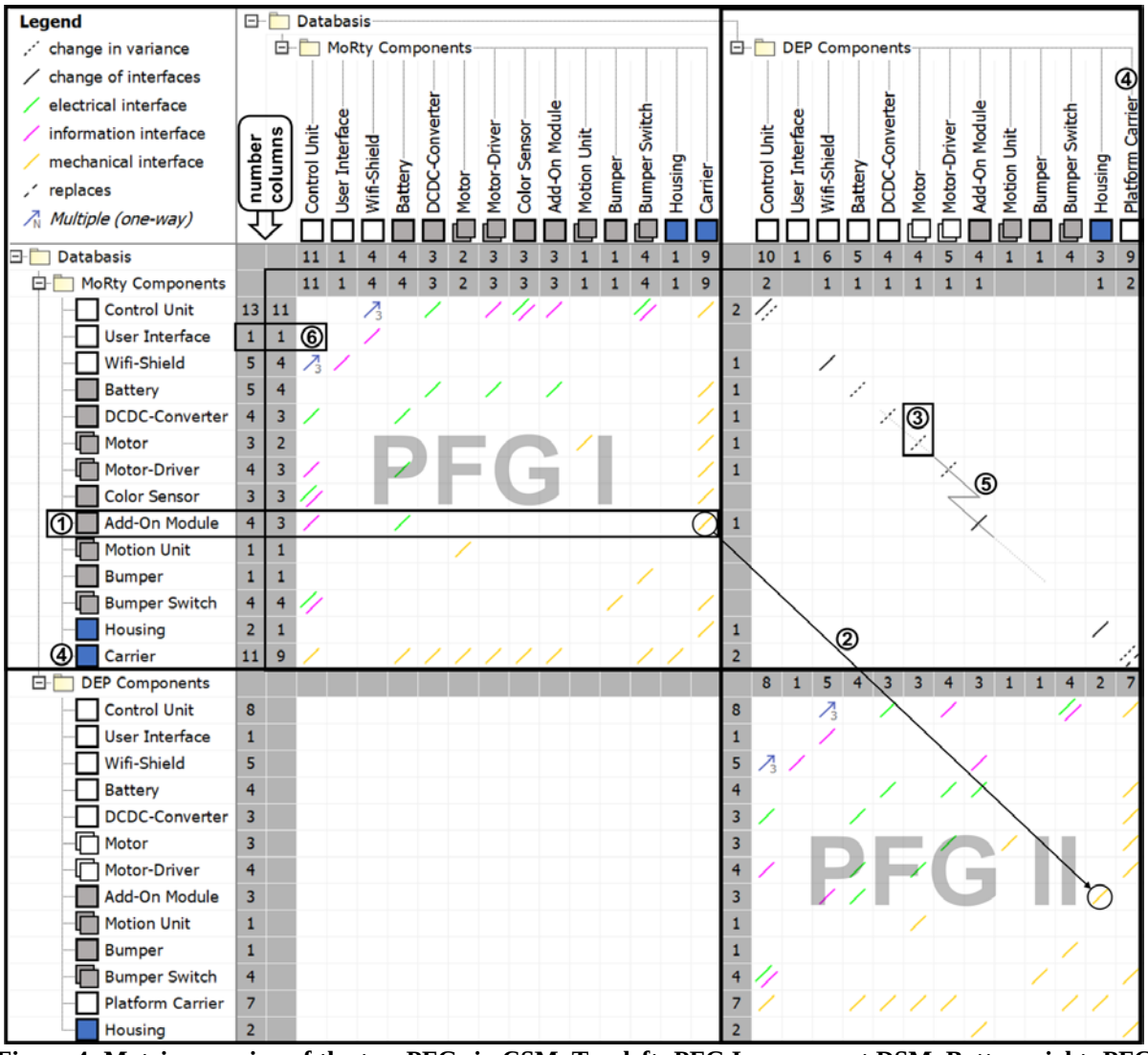

Figure 4: Matrix mapping of the two PFGs in CSM. Top left: PFG I component DSM. Bottom right: PFG II component DSM. Top right: Delta-Matrix.

In the Delta-Matrix, the changes made between the generations are documented and can be traced. Whether a change has taken place at all can be seen in the number columns, comparing the numbers in the number columns on the left side of the matrix. The difference between the numbers in one line shows how many changes have occurred. Possible changes are change in variance, change in interfaces and replacements. In this section examples of the different cases are shown. In PFG I the *housing* had only one mechanical interface to the c*arrier*. In PFG II there is another mechanical interface to the *add-on module* as it was removed from the bottom (interface to carrier in PFG I) and added to the top (interface to housing in PFG II) as can be seen in Figure 4. This change can be assigned to the *attaching/ removing* mechanism explained in 2.1. and can easily be comprehended, comparing the matrix entries (we see a shift of the entry, ②). An example for a change of the variance is the *motor* component. In PFG I the *motor* was still conceived as a variant, in PFG II it is a standard part, what results from the new development objectives. The difference can also be seen in the icons that are linked to the components ③. A replacement took place once: The *carrier* was largely redesigned and became the *platform carrier* in PFG II ④, which includes all components which where standardised due to the new objective. In addition to the three change types, which are listed in the Delta-Matrix, further information can be taken from Figure 4. An indication for the removing of a component can be seen in the changing shape of the delta-matrix (number of rows > number of columns). In our case, the colour sensor was removed from PFG I to PFG II. This shifts the diagonal downwards ⑤. The same can be done in reverse for the addition of components. If a component has not changed from PFG I to PFG II there is no entry in the Delta-Matrix and the numbers in the number columns are the same ⑥.

4 Discussion & Outlook

In this paper different tools were presented to show the data correlations. The presentation of a network graph (Figure 2) allows to quickly uncover critical elements and paths. In addition, elements that belong together are grouped and the mutual influence becomes obvious. The limits lie in the complexity of the relationships, which increases with the complexity of the product structure under consideration. The representation of data relationships in a DSM is then clearer. In addition, differences in the structures can be identified more quickly by comparing matrix entries (Figure 4). The combination of different representations of the data correlations seems to make sense to tackle the problem from different perspectives. However, it should be noted that the use of different software can lead to a loss of traceability.
Since the PFGs are a real development case for an education course (Heyden et al. 2020), the know-how about the set development objectives and the changes made within the product architecture is high. The product architecture as well as the overall system structure are not complex enough to benefit from higher graph analysis at this point, but the relatively small matrices in all domains are regarded as helpful to comprehensively highlight the change matrix and overall system effect structure. The product architecture as well as the overall system structure are not complex enough to benefit from higher graph analysis at this point, but the relatively small matrices in all domains are regarded as helpful to comprehensively highlight the change matrix and overall system effect structure. Especially the connection from development objectives, their influence on product features and their effect on component variety in terms of the expanded and standardised platform module and the subsequent possibility of standardising the mechanical and electrical interfaces. With the approach shown, the planning of further PFGs can be supported by extending by the generation MDMs with a further PFG MDM. Not only the differences between PFGs can be shown. Other systems can also be compared, such as two different product variants of a product family or two product

alternatives. An application for the iterations between digital twins and real product variants is also conceivable with this approach.

References

Albers, A., Bursac, N., & Rapp, S. (2016). PGE - Product Generation Engineering - Case Study of the Dual Mass Flywheel. International Design Conference - DESIGN 2016, Dubrovnik - Croatia, May 16 - 19, 2016., 791–800.

Browning, T. R. (2016). Design Structure Matrix Extensions and Innovations: A Survey and New Opportunities. IEEE Transactions on Engineering Management, 63(1), 27–52. https://doi.org/10.1109/TEM.2015.2491283

Bursac, N. (2016). Model Based Systems Engineering zur Unterstützung der Baukastenentwicklung im Kontext der Frühen Phase der Produktgenerationsentwicklung (Dissertation). Karlsruher Institut für Technologie, Karlsruhe. https://doi.org/10.5445/IR/1000054484

De Weck, O.L. (2007). On the role of DSM in designing systems and products for changeability. On the role of DSM in designing systems and products for changeability. International DSM Conference, Munich, Germany, 16.-18.10. (pp. 311-324).

Du, X., Jiao, J., & Tseng, M. M. (2001). Architecture of Product Family: Fundamentals and Methodology. Concurrent Engineering, 9(4), 309–325. https://doi.org/10.1177/1063293X0100900407

Ehrlenspiel, K., & Meerkamm, H. (2013). Integrierte Produktentwicklung: Denkabläufe, Methodeneinsatz. München: Hanser. https://doi.org/10.3139/9783446436275

Erixon, G. (1998). Modular Function Deployment - A Method for Product Modularisation (Dissertation). The Royal Institute of Technology, Stockholm.

Feldhusen, J., & Grote, K.-H. (2013). Pahl/Beitz Konstruktionslehre. Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-29569-0

Heyden, E., Küchenhof, J., Greve, E., & Krause, D. (2020). Development of a Design Education Platform for an Interdisciplinary Teaching Concept. CIRP Design Conference. South-Africa, 2020.

Holt, J., & Perry, S. (2014). Sysml for systems engineering: A model-based approach (2. ed.). Professional applications of computing series: Vol. 10. London: Inst. of Eng. and Tech..

Kipp, T. (2012). Methodische Unterstützung der variantengerechten Produktgestaltung (Dissertation). Technische Universität Hamburg, Hamburg.

Krause, D., & Gebhardt, N. (2018). Methodische Entwicklung modularer Produktfamilien: Hohe Produktvielfalt beherrschbar entwickeln. Berlin: Springer Verlag. https://doi.org/10.1007/978-3-662-53040-5

Küchenhof, J., & Krause, D. (2019). Entwicklung eines Produktarchitekturmodells zur Ableitung modularer Produktstrukturen. In Proceedings of the 30th Symposium DFX, 2019. https://doi.org/10.35199/dfx2019.3

Küchenhof, J., Schwede, L.-N., Hanna, M., & Krause, D. (2019). From Visualizations to Matrices – Methodical support for New Development of Modular Product Families. 21st Int. DSM Conference, Monterey, USA, 2019. https://doi.org/10.35199/dsm2019.11

Küchenhof, J., Tabel, C., & Krause, D. (2020). Assessing the Influence of Generational Variety on Product Family Structures. CIRP Design Conference, South-Africa, 2020.

Lindemann, U., Maurer, M., & Braun, T. (2009). Structural Complexity Management: An approach for the field of product design. Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-87889-6

Orawski, R., Hepperle, C., Schenkl, S., Mörtl, M., & Lindemann, U. (2012). Life-Cycle Oriented Requirement Formalization and Traceability. IFIP Advances in Information and Communication Technology. Product Lifecycle Management. Towards Knowledge-Rich Enterprises (Vol. 388, pp. 124–133). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-35758-9_11

Sanchez, R. (1996). Strategic product creation: Managing new interactions of technology, markets, and organizations. European Management Journal, 14(2), 121–138. https://doi.org/10.1016/0263-2373(95)00056-9

Scherer, H., & Albers, A. Model Based Methods for the Modeling of the System of Objectives and the Correlation between Form and Function to support the Series Development of Modular Systems Using the Example of Hybrid Powertrains. (Dissertation). Karlsruher Institut für Technologie, Karlsruhe.

Stone, R. B. (1997). TOWARDS A THEORY OF MODULAR DESIGN (Dissertation). The University of Texas at Austin, Texas.

Weilkiens, T. (2008). Systems engineering mit SysML/UML: Modellierung, Analyse, Design. - Title from resource description page (viewed May 11, 2009) (2. Aufl.). Safari Books Online. Heidelberg: dpunkt-Verlag.