GENERATING A META-MODEL FOR MODULARIZATION METHODS

J. Hackl, T. Gumpinger and D. Krause

Keywords: product family, modularization, multiple-domain matrix, product structure

1. Introduction
Product architecture has to reflect the needs of the lifecycle phases of a product. Megatrends like globalization and individualization change these needs and new demands on product architecture arise. These have to be addressed to be competitive in a changing market. Producing individualized product variants is a major challenge in satisfying market demands. Modularizing product structures is a typical way to achieve this; several product development methods have been developed and applied in industry. They can be classified by their modularization goals, whether technical-functional or product-strategic. These methods need different information for their application. This paper analyses commonly applied methods for their data acquisition needs and information processed by their domains and relations used. This information is used to synthesis a generic information framework for modularization. Seven methods are analysed using the approach presented in the next section. The steps of each approach are described and domains and relations used are identified. The information acquired is subsequently processed into an overall information framework and evaluated. Therefore, the previously used author-specific domains and relations are generalized at an abstract level. The methods’ information schemes are united into a common meta-model.

2. Analysis Model
An abstract form is chosen to compare different modularization methods. In this paper the processed information is of interest, hence the generation and flow of information needs to be represented. Multiple-domain modelling [Maurer 2007] is a generic way to describe the information of complex systems. The general idea of multiple-domain matrices (MDM), represented by domains and relations, is adapted to represent the information needs and flow of the methods. Domain means a kind of entity representing a specific view [Kreimeyer and Lindemann 2011]. Relations are the interactions or links between the domains or within a domain. By collecting the domains and their relations it is possible to get a more abstract view of the methods. This is necessary to find similarities and differences. Kreimeyer and Lindemann [2011] use a similar approach to compare several methodologies in the field of structural process modelling.

The methods are examined in three steps. First, a short description of the method is given, within which the goals and the typical initial situation of the method are described. Thus it is possible to find out if methods with the same goal use a similar approach by connecting the same domains with similar relations. The initial situation might play a role in how a method is set up.

The methods are classified into the underlying scheme of Suh [2005]. Suh created a concept of domains which was used to build an axiomatic design framework: a method for a structured design of systems. According to Suh the design world consists of the four domains customer, functional,
physical and process. He states that many systems of product design can be described using them. Simpson et al. [2006] adds logistics to Suh’s domains, summarising the domains into three parts: front-end issues, optimization-based methods and back-end issues. Figure 1 shows the simplified scheme of Simpson, which will be used to group the methods, facilitating the comparison of methods. After classifying the methods in this scheme, individual method steps can be allocated to the area in which they provide solutions and can be specifically chosen.

The following section contains a summary of every method. The information used is transferred into the domain space. For example, components relate to the physical domain and functions to the functional domain. Any attributes (written in italics), like costs of components, which specify domain elements are recorded in the step description below each MDM. The relations between domains are located in the fields of the matrices. Characters within the fields enumerate the sequence of the steps. Specific relation attributes are also in the step description. The second classification is in the field of knowledge management and concerns the terms data, information, knowledge, wisdom [Rowley 2007]. The key parts of the methods are found by analysing the steps in which data are acquired, information and knowledge is created and new insights are gained. This results in a content quality of the methods steps. Each method closes with a short summary.

3. Analysis of methods for a modular product architecture

In this section, seven methods for developing modular product structures are analysed, as described in the previous section. This selection of methods provides initial insights into the information needed for modularization. Krause gives an overview of further methods to be evaluated using this approach [Krause and Ripperda 2013].

Hölttä-Otto – Modular product platform design

Hölttä-Otto [2005] extends existing modularization methods for optimizing product families by considering variants. The approach encompasses three metrics. The first is flexible interface design, which lets the user identify critical interfaces in a product architecture. The main approach lies in evaluating the main flows between functions and their effort when being changed. The second metric is the identification of common modules across the product family, which is done by normalizing all inputs and outputs of product functions and calculating the Euclidian distances between functions. The same is possible for the physical domain, using component requirements and attributes like weight, for example. The Euclidian distances are then drawn in a dendrogram to derive an adequate product structure. The various platform concepts are evaluated with the third metric. Figure 2 shows the MDM that can be developed from Hölttä-Otto's metrics.

<table>
<thead>
<tr>
<th>Function</th>
<th>Component</th>
<th>Module (Platform concept)</th>
<th>Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>A. Coupled with Flow</td>
<td>C. Summed up to</td>
<td></td>
</tr>
<tr>
<td>Component</td>
<td>B. Change effort</td>
<td>D. Euclidian Distance</td>
<td></td>
</tr>
<tr>
<td>Module Platform concept</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Strategy</td>
<td></td>
<td>E. Scored</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2. Abstracted MDM as per the method of Hölttä-Otto [2005]
Table 1. Method steps of Hölttä-Otto [2005]

<table>
<thead>
<tr>
<th>Step</th>
<th>Attribute</th>
<th>Quality of content</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Building of functional structure</td>
<td>Flow</td>
</tr>
<tr>
<td>B</td>
<td>Evaluation of change effort of flow types</td>
<td>Change effort</td>
</tr>
<tr>
<td>C</td>
<td>Evaluation of modules regarding change effort sum</td>
<td>-</td>
</tr>
<tr>
<td>D</td>
<td>Deriving of interfunctional/inter-component Euclidian distance; alignment in dendrogram</td>
<td>Euclidian distance</td>
</tr>
<tr>
<td>E</td>
<td>Scoring of platform concepts</td>
<td>-</td>
</tr>
</tbody>
</table>

Hölttä-Otto’s metrics concentrate on the two domains of function and component. They are specifically designed for the field of product family design, relating to Simpson. Her metrics are of a technical-functional nature. Hölttä-Otto communicates the product family’s degree of commonality by visualizing the optimal module fragmentation with a dendrogram.

Göpfert – Modular product families

Göpfert’s [1998] Metus method aims to harmonize product structure and organisational structure by developing a modular product architecture. This contributes to a reduction in organization efforts in the product development process. The initial situation for Metus deployment is typically an existing product family, where the external variety needed has often led to an increase in internal variety. With an academic background, this method is transferred into a consulting service. Metus organizes components into modules that consider the functional aspect and the organizational aspect. The MDM below (Figure 3) represents this in an abstracted context.

![Figure 3. Abstracted MDM as per the method of Göpfert [1998]](image)

Table 2. Method steps of Göpfert [1998]

<table>
<thead>
<tr>
<th>Step</th>
<th>Attribute</th>
<th>Quality of content</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Building of functional structure</td>
<td>-</td>
</tr>
<tr>
<td>B</td>
<td>Mapping of functions to components</td>
<td>-</td>
</tr>
<tr>
<td>C</td>
<td>Building of modules</td>
<td>-</td>
</tr>
<tr>
<td>D</td>
<td>Organizational structure</td>
<td>-</td>
</tr>
</tbody>
</table>

Göpfert reorganizes components into modules, benefitting the domains that influence this aggregation. It contributes to the functional binding and organizational structure. He enhances his method with the metus rhombus visualization. This visual representation relates two domains and groups them hierarchically.

Jiao – A methodology of developing product family architecture for mass customization

Jiao et al. [1999] generally aim to increase product variety in mass customization while keeping development and production efforts as lean as possible. Keeping manufacturing costs low is another
aim mentioned. The initial situation is an existing mass product with increasing external variety, which leads to an increase in internal variety. The method utilizes systematic steps to formulate the product family architecture. Jiao analyses the problem from three different views: functional, physical and technical. He develops a new product structure by going through all of Suh’s [2005] domains, from customer requirements to functional requirements, design parameters and components to the final product structure, considering cost and performance.

<table>
<thead>
<tr>
<th>Step</th>
<th>Attribute</th>
<th>Quality of content</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Formulation of functional requirements (FR)</td>
<td>- Data acquisition</td>
</tr>
<tr>
<td>B</td>
<td>Hierachring FRs down to instances/variables</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>Customer exploration</td>
<td>No. planned products instance and importance</td>
</tr>
<tr>
<td></td>
<td>Most important FR instances across customers</td>
<td>- Customer knowledge necessary</td>
</tr>
<tr>
<td></td>
<td>Similar FR instances across customers</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>Mapping of FR to Design Parameters (DP)</td>
<td>- Data acquisition</td>
</tr>
<tr>
<td>E</td>
<td>Module generation; clustering FR-DP matrix</td>
<td>- Information about design modules and technical modularity</td>
</tr>
<tr>
<td>F</td>
<td>Component determination regarding design modules and process capabilities</td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>Performance of modules</td>
<td>performance</td>
</tr>
<tr>
<td></td>
<td>- Information about performance gained</td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>Cost estimation</td>
<td>cost</td>
</tr>
<tr>
<td>I</td>
<td>Final configuration structure after economic evaluation in G. and H.</td>
<td>- Previous information and knowledge needed</td>
</tr>
</tbody>
</table>

Jiao creates a modularization approach which he reconciles for technical performance, costs and process capabilities. He presents a broad method that addresses and assesses all domains except logistics.

**Lindemann – Structural Complexity Management**

With structural complexity management, Lindemann et al. [2009] offers a systematic approach for managing complex structures in product design. It is not specially made but is applicable to the development of modular product structures. The approach leads from system definition and information acquisition to the deduction of indirect dependencies, structure analysis and to an improved product system. For modularization he finds the key domains components, functions and features, which can be extended to a specific problem.
Multiple-domain modelling is a generic approach for structural complexity management. It is closely related to the Design Structure Matrix (DSM) and graph theory. Hence many clustering algorithms are available for modularization. Software tools, like LOOMEO, support the approach and provide a visual representation of the MDMs using node-link diagrams. A powerful possibility is the deduction of indirect dependencies of domains. This gives insights into previously unknown relationships. Multiple case studies use the approach for modularization of products.

**Pimmler/ Eppinger – Design Structure Matrix (DSM)**

Pimmler and Eppinger [1994] adapt the DSM [Steward 1981] with the aim of improving the product architecture and team structure. It is a technical-functional modularization that clusters the functional or component domain and is carried out to redesign single products. The method’s main idea is the breakdown of a product into components or functions (in the early development phase). These are grouped into modules based on coupling criteria that quantify the strengths between the components. A cluster algorithm swaps rows and columns to regroup the DSM elements.

### Table 4. Method steps of Lindemann et al. [2009]

<table>
<thead>
<tr>
<th>Step</th>
<th>Attribute</th>
<th>Quality of content</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Flow-oriented functional model</td>
<td>- Data acquisition</td>
</tr>
<tr>
<td>B</td>
<td>Record geometric dependencies</td>
<td>- Data acquisition</td>
</tr>
<tr>
<td>C</td>
<td>Record of direct dependencies</td>
<td>- Data acquisition</td>
</tr>
<tr>
<td>D</td>
<td>Deduction of indirect dependencies</td>
<td>- Information generation about indirect dependencies of components</td>
</tr>
<tr>
<td>E</td>
<td>Structure analysis by clustering all component related matrices</td>
<td>- Aggregated information for module candidates</td>
</tr>
<tr>
<td>F</td>
<td>Generation of product structure depending on relations</td>
<td></td>
</tr>
</tbody>
</table>

### Table 5. Method steps of Pimmler and Eppinger [1994]

<table>
<thead>
<tr>
<th>Step</th>
<th>Attribute</th>
<th>Quality of content</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Decomposition of product</td>
<td>- Product knowledge needed</td>
</tr>
<tr>
<td>B</td>
<td>Coupling between components</td>
<td>spatial... - Data acquisition</td>
</tr>
</tbody>
</table>
The consideration of couplings between domain elements allows a deeper understanding of the technical-functional dependences of the modular product structure. The influence of the clustering algorithm has to be considered. Pimmler and Eppinger use a holistic algorithm that centres the positive quantified elements at the main diagonal of the DSM.

**Simpson – Integrated approach to product family design**

Simpson et al. [2012] takes several detached methods and integrates them. He develops communal components by translating customer requirements into commonality specifications. The aim is to find a product platform for the analysed product family.

<table>
<thead>
<tr>
<th>Step</th>
<th>Attribute</th>
<th>Quality of content</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Mapping of variants to market segments and price/performance tiers</td>
<td>- Information about assigning products to market segments</td>
</tr>
<tr>
<td>B</td>
<td>Mapping customer requirements (CR) to engineering requirements (ER)</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>Mapping of ER to components</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>Rating ER – component matrix on <em>Likelihood of redesign</em></td>
<td>- Knowledge of future changes needed, information about candidates for a communal platform outcome</td>
</tr>
<tr>
<td>E</td>
<td>Deriving a coupling index</td>
<td>- Information about component coupling</td>
</tr>
<tr>
<td>F</td>
<td>Derivation of a commonality index</td>
<td>- Information about commonality of the product family</td>
</tr>
<tr>
<td>G</td>
<td>Multi-Objective Optimization</td>
<td></td>
</tr>
</tbody>
</table>

In the approach shown he uses existing modularization methods that can be exchanged as needed. The approach especially looks at the variety across variants. The framework overcomes the fragmentation of the individual methods used.
Krause – Integrated PKT-approach

The integrated PKT-approach by Krause and Eilmus [2011] aims to reduce internal variety while maintaining external variety levels for the customer. The initial situation is an existing product family that has to be redesigned due to an increase in variety. Krause looks at both technical-functional modularization and product-strategic modularization. The approach consists of two major parts. First, the product family is redesigned to comply with design for variety [Kipp 2012]. The components of the adapted product family are then clustered according to the entire product lifecycle [Blees 2010].

<table>
<thead>
<tr>
<th>Step</th>
<th>Attribute</th>
<th>Quality of content</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Specifying customer relevant properties with variant values</td>
<td>- Data collection</td>
</tr>
<tr>
<td>B</td>
<td>Mapping of variant values to variants</td>
<td>- Information about variety in the product family</td>
</tr>
<tr>
<td>C</td>
<td>Analysing functions, working principles and components on the attributes</td>
<td>Standard, variant, optional</td>
</tr>
<tr>
<td>D</td>
<td>Allocation of the variant elements of domains into A - C</td>
<td>- Data collection</td>
</tr>
<tr>
<td>E</td>
<td>Generation of solutions for design for variety</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>Clustering of components according to flow schemes</td>
<td>- Module definition for technical-functional modularization</td>
</tr>
<tr>
<td>G</td>
<td>Developing modularizations for every life phase relevant for the company</td>
<td>- Information about conflicts between the life phases</td>
</tr>
<tr>
<td>H</td>
<td>Harmonizing the modularizations to a final concept</td>
<td>- Expert knowledge necessary for finding an optimal solution</td>
</tr>
</tbody>
</table>

This approach for developing modular product families is workshop-oriented and therefore uses many visualization tools to communicate the results to the customer. Commonly used methods are utilized for modularization. The step of harmonizing different modularizations of a product family across its lifecycle differs from other methods.
4. Evaluation of the methods and derivation of a meta-model
To evaluate the methods, their information flows need further abstraction. The author-specific domains were previously assigned to the MDM. To unify them, the domains are sorted into the scheme by Suh and Simpson (Figure 1). The general domains in this figure are color-coded. For reference, these colours are reused in each method’s MDM (Section 3) to indicate the domain affiliation, e.g. Lindemann et al [2009] describe component and feature domains that both correspond to the physical domain. The relations of the author-specific domains are reordered by this classification scheme.

This sorting creates the possibility of combining the methods into an overall MDM (Figure 9). The appearance of relations between domains is counted within the fields. All methods use a relation within the physical domain and nearly all use one in the functional domain as well. This can be interpreted, as most of the analysed methods approach the modularization problem from the technical-functional point of view [Krause and Ripperda 2013]. Modularization eventually has to affect the physical domain, grouping components into modules.

In addition, the relations count from the functional domain to the physical domain is distinctive. This reflects that mapping of functions to components or modules is a basic concept of modularization (function binding in [Salvador 2007]). The domains surrounding the functional and physical centre domains are less connected. The customer domain has a big influence on modularization methods. This is represented by the active sum (5), the sum of values in each row, which indicates the overall influence a domain has on other domains [Lindemann et al. 2009]. The passive sum, sum of values in each column, shows how influenced a domain is. Figure 9 (right) illustrates the analysed methods’ domains with an influence portfolio, mapping active to passive sum. Again, functional domain and physical domain are highly interactive.

<table>
<thead>
<tr>
<th>Customer Domain</th>
<th>Functional Domain</th>
<th>Physical Domain</th>
<th>Process Domain</th>
<th>Logistics Domain</th>
<th>Active sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer Domain</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Functional Domain</td>
<td>1</td>
<td>6</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Physical Domain</td>
<td>1</td>
<td>2</td>
<td>7</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Process Domain</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Logistics Domain</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Passive sum</td>
<td>3</td>
<td>11</td>
<td>16</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 9. Relationships among domains across all analysed methods; influence portfolio

Relations with the same meaning across the methods’ MDMs can be unified. The matrix above is extended with these relationships. This gives an information framework for modularization of product families. Further detailing of the domains and relations with attributes results in a meta-model (Figure 10). Unified relations with a high occurrence are marked in bold. Relations are important because they enable aspects of the modularization to be put in context. The context gives new insights and enables matching with the product structure. For example, Pimmler/Eppinger optimized the team structure by relating the process domain (persons) with the physical domain (components). Methods try to create domain-specific benefits by relating and harmonizing structures of or within a domain.

The coupling elements of the functional domain (by assigning relations with flow-based attributes) occur several times within the analysed methods. Especially in early design phases, but without a physical structure, this abstraction is used to cluster functions into modules. When modularization is applied to existing product families the couplings of the physical domains are analysed. Some methods specify the couplings with attributes (e.g. flow types). Most of the analysed methods are only partly product-strategic. Hence the derived meta-model preferably represents technical-functional modularization.
Methods that pursue the same aim were analysed by comparing their steps and required information flows. Göpfert and Pimmler/Eppinger aim for the alignment of product structure and team structure. Pimmler and Eppinger analysed couplings of functions or components within a domain and clustered them with an algorithm. Göpfert connects functions and components, and aligned the subsequent functional structure with the component structure to derive modules across domains. He matches the organizational structure across domains to functional and physical elements, where Pimmler/Eppinger derive the team structure from the modularization.

Jiao and Krause aim to maintain external variety levels while reducing internal variety. Both modularize from the technical-functional and the product-strategic view. Jiao builds a functional structure from functional requirements, leading to instances of the function. Krause specifies the customer domain (customer relevant properties) by relating the functional domain (variant values). In application, customer requirements and functional requirements often correspond; hence both methods need similar information. The step of module definition differs between the methods. Jiao clusters the functional requirements with design parameters and evaluates technical performance and cost of the modules. Krause uses design for variety methods to form components, which he modularizes for each product life phase and then harmonizes across the phases. That difference may have its cause in the workshop-based approach of Krause, which is designed for use in the corporate context, while Jiao’s approach is of an analytical nature.

5. Conclusion and Outlook
In this paper an information framework was derived by extracting domains, relations and their attributes from different modularization methods. The methods’ information flow was traced on an abstract level using an MDM. This revealed the underlying scheme of each method, which enables comparison of different methods in a concise way. By combining method schemes a common meta-model can be extracted. Its evaluation indicated the core domains for modularization: physical and functional domains. Keeping in mind that most of the considered methods are technical-functional in nature, this study should be extended to include more modularization methods. Available case studies on the methods can be incorporated to sharpen their profile. A comparison of modularization theory and practical application using this approach could bring further insights.

Different sets of methods were developed, approaching modularization from different perspectives. Many methods and tools exist in isolation from each other [Simpson et al. 2012]. This framework...
fosters the combined use of methods by being transparent about their information needs. This is of special use in methodical toolkits like the integrated PKT-approach [Simpson 2014]. Information interfaces between method units for case-specific adaption of methodical tool chains can be identified. An information database for a methods toolkit can be implemented. The unifying nature of this approach contributes to consolidation of modularization methods.

References