INTERNATIONAL DESIGN CONFERENCE - DESIGN 2006
Dubrovnik - Croatia, May 15 - 18, 2006.

DESIGN 2006

USING MULTIPLE DESIGN STRUCTURE MATRICES

M. Eichinger, M. Maurer and U. Lindemann

Keywords: design structure matrix, product model, design process

1. Introduction

In product development, multiple aspects of engineering design methodology are integrated and interact with each other. For example, functions depend on components that realize these functions, and features of a product depend on product functions. The different aspects of product development are called "domains" in the following. To depict and optimize the structure of these domains, different methods have been developed. The Design Structure Matrix (DSM) [Steward 1981] is applied to structure-, task-, organization-, and parameter analysis [Browning 2001] and offers methods [Baldwin & Clark 2000; Kusiak 1999] for a concise visualization of complex network structures. However, the existing approaches are lacking a holistic view onto the domains of product design as they consider the domains only independent from each other, e.g. only components and their relations are modeled in a DSM. A method that relates two different domains (for example components and functions) is given by the Domain Mapping Matrix (DMM) [Danilovic & Browning 2004] or the K&V-Matrix [Bongulielmi et al. 2001]. However, these approaches are limited to two domain combinations only. Other important methodologies that use interdomain-matrices are Axiomatic Design [Suh 2001], QFD [Hauser & Clausing 1988]. Some of the methodologies (element-level matrices) are classified in Malmqvist [2002] into intra- and inter-domain matrix methodologies. We adapt this specification to define the Multiple Design Structure Matrix that integrates intra- and inter-domain matrices. This methodology offers the possibility to analyze sets of multiple domains and derive further insight into the engineering design process by evaluating cross-domain relations. Although the coherent representation form offers new benefits for product design optimization, it also leads to big matrices that have to be filled and analyzed. To avoid costly filling processes and tedious matrix analysis, we developed an algorithm that points out important dependencies within the matrix and helps the designer to identify important parts in a MDSM using cross-domain relations between the different MDSM domains. First, we briefly describe the MDSM methodology for product design optimization and then present an algorithm for the evaluation of the cross-domain relations.

2. Combining Multiple Product Design Domains

Figure 1 shows a general MDSM structure, consisting of a symmetric alignment of elements on both axes and element groups of different domains. This formation causes sub-matrices of DSM and DMM types. The sub-matrices aligned along the MDSM diagonal are DSMs, the sub-matrices in the upper and lower triangular of the MDSM are consequentially DMMs. In a MDSM, bidirectional relationships are modeled, that is, the matrices in the upper and lower triangular of the MDSM do not necessarily contain the same information. For isolated matrices, analysis algorithms exist, e.g. partitioning, clustering or loop identification for DSM structures [Baldwin & Clark 2000; Kusiak 1999]. Analyzing the connectivity of isolated structure-, task-, organization-, or parameter-matrices may offer valuable insight; however, most of the problems in engineering design are multidisciplinary problems that have to be addressed as a whole. So far, no algorithms exist for a holistic analysis of the integrated domains in MDSM structures.

To enable such an analysis, a set of product design domains has to be defined first. The following set has turned out to be useful for MDSM optimization as it covers a broad range of product design elements:

- Components (assemblies or parts of a product that are decomposed to a defined level of detail)
- Functions (desired or non-desired behavior of the product, assemblies or components)
- Parameters (settings that usually apply directly to components)
- Resources (objects and people needed to develop the product)
- Tasks (specific, definable activities to perform an assigned piece of work, often finished within a certain time)

In practice, the set of domains for MDSM analysis has to be adapted to the specific problem the designer has to examine. However, we used this set of domains during several MDSM optimization procedures and found that it is appropriate for most problems in engineering design, as it integrates multiple views onto the problem. Additional research is needed to examine the implications of hierarchical dependencies across multiple domains (i.e. what implications have to be considered along a development chain such as requirements-functions-specification-components-parameters-tasks). So far, the MDSM is intended to depict and optimize the as-is-state of a product development issue.

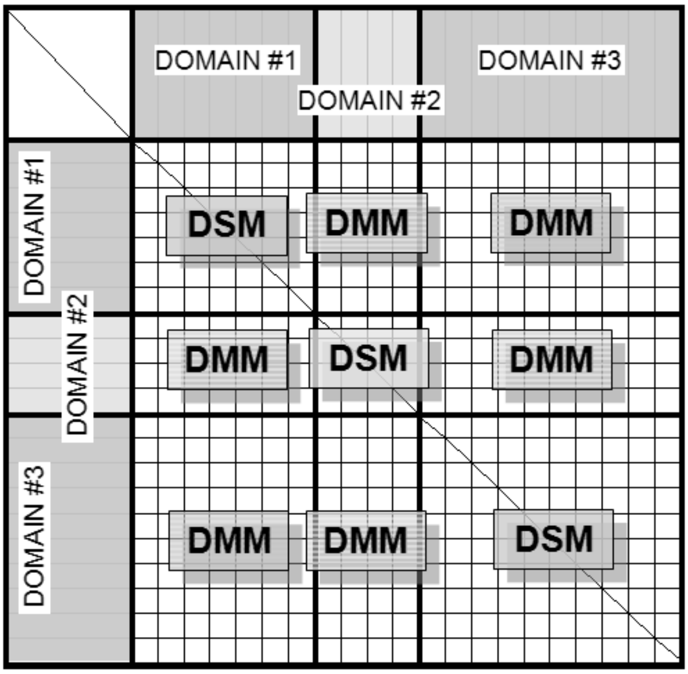

Figure 1. Integrating multiple domains into one MDSM

The MDSM contains all intra- and inter-domain relations between the included domains. For example, the mutual relations between components and functions are modeled in the components-functions (directed impact from components to functions) matrix and the functions-components (directed impact from functions to components) matrix. For each domain combination two matrices exist that represent the directed relation between these domains. The MDSM relations (the content of the MDSM) are not necessarily symmetrical to the diagonal of the matrix, even if some sub-matrices may be symmetrical (a matrix that describes physical relations between components, for example, is always a symmetrical matrix due to the principle that actio equals reactio). The aggregation of the product domains in a MDSM offers new possibilities of interdependency analysis between the domains. These have not been considered so far in method based matrix handling. Analyzing and visualizing these interdependencies is quite important, as these cross-relations may be the reason for undesired or unpredictable product or process behavior. The cross-relations mostly span one domain (for example, two people are related through a common component they work on). In the following, we propose an approach for evaluating these cross-domains relations.

3. Multi-Domain Product Design Analysis

3.1 Problem description

Only direct relations are allowed in a DSM in order to receive helpful information. These relations are usually obtained during one or several team workshops. Experiences with matrix filling processes indicate that the distinction between indirect and direct relations is a huge problem in the perception of product design structures. Indirect relations are defined as relations that are caused through a dependency chain that spans one or more elements.

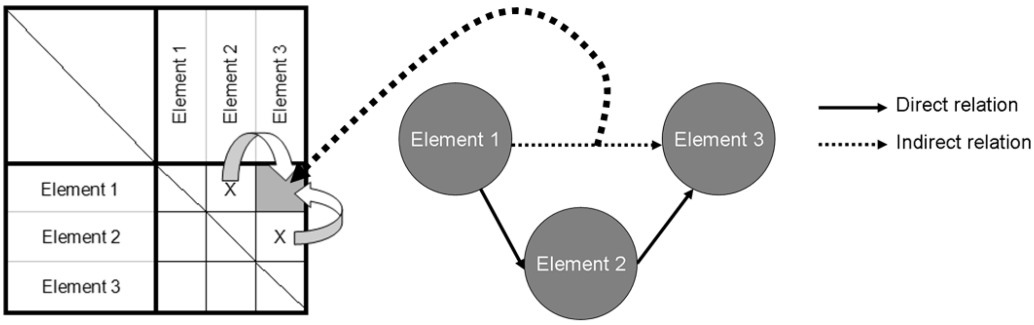

Figure 2. Direct and indirect relations between matrix elements

Team members that take part in a matrix filling process often are not able to distinct between direct and indirect relations, as the underlying structure is intuitively known (mostly by experience) or largely unknown. A chain that spans one or more elements (figure 2) may lead to a false perception of the real product structure. The designer may only know that two elements are “somehow” related and therefore place the mark in the wrong matrix cell (see figure 2) to avoid the detour over the intermitting element. Finally, the captured matrix structure would not represent the real structure and therefore the quality of analyses would decrease.

With multiple domains, this problem gets even worse, as indirect relations may also be caused through relations across different domains of the matrix. These indirect relations are even harder to identify, because the designer has to compare different contexts and meanings of elements. However, multiple domains also offer new possibilities for the analysis of cross-domain relations. We found that these domain-spanning indirect relations are very important in practice, as they are responsible for the distinction between direct and indirect relations within a domain. The team that conducts the MDSM analysis may e.g. see a direct relation between two persons if there are many other elements connecting these persons (like common components, common tasks or common data resources). The same team may, however, deny a direct relation between the persons if there is e.g. only one unimportant data resource these persons share. The here presented algorithm offers the possibility to focus on important elements within the matrix that are connected through multiple other matrix elements.

Another problem with DSM and MDSM analysis is the size of the matrices. MDSMs can get very big if several domains are integrated into a single matrix. Given a MDSM with five domains and ten elements per domain, the MDSM is a 50x50 matrix with 2,450 relations to be evaluated. If used during a matrix filling process, the here presented algorithm is able to visualize important relations and helps the team filling the matrix more quickly.

3.2 Calculation algorithm

For accessing information about direct and indirect linking, we propose an algorithm that uses data already stored in the matrices in order to determine the indirect relations between elements. These indirect relations may then be visualized to help the designer identifying relations that should be checked more precisely. We only consider indirect relations that span one element (like in the example in figure 2). Of course, longer chains of indirect interdependencies exist; however, information gain for designers is not relevant as they cannot perceive longer chains of elements properly due to the high density of interrelations in common structures.

The analysis process we propose is called “AID-Process” (Analyzing Indirect Dependencies-Process). The algorithm determines the indirect relations between two elements of the same or different domains by using the adjacent matrices that compose the MDSM. These indirect dependencies are then visualized and help the designer identifying dependencies between elements that are caused through indirect relations in other domains. This visualization can help a designer to speed up the matrix filling process and to ease the identification of important matrix elements. If for example the indirect relations between components and functions are to be evaluated, the matrices of interest are the components-parameters and parameters-functions matrices (figure 3). The AID-Process is merely used for the analysis of existing specifications, products and processes. It is therefore not necessary to sort the domains according to an overall logic or abstraction level, as it is not intended to obtain for example a task sequence from a requirement-, functions- or components-structure.

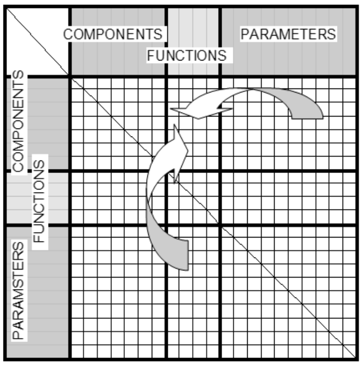

Figure 3. Exemplary sources for the evaluation of indirect relations

Depending on the total number of domains, there may be more than one matrix pair to be evaluated. Taking the suggested domains from above (components, functions, parameters, resources, and tasks), there would be also the components-resources and components-tasks pairs that contain cross-domain relations. The AID-Process can be executed in filled and partly filled MDSMs and proceeds as depicted in figure 4.

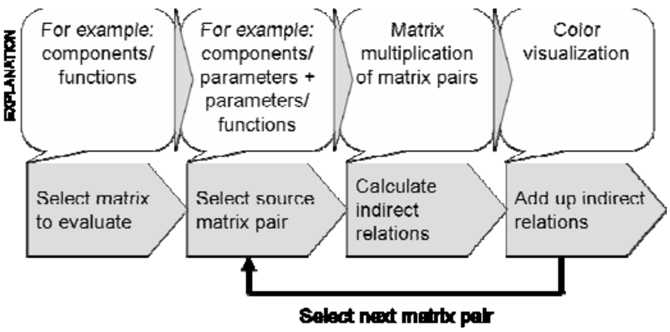

Figure 4. AID-Process

The process starts with selecting the MDSM sub matrix (representing e.g. a product from a specific domain view) that is to be evaluated. In a second step, the adjacent sub-matrix pairs that act as the source for the evaluation of indirect relations are determined. If for example the components-functions matrix has been selected for evaluation, a possible adjacent matrix pair is the components-tasks and tasks-functions matrix. As depicted in figure 5, there is always a third domain connecting two matrices (domain "B" in the example).

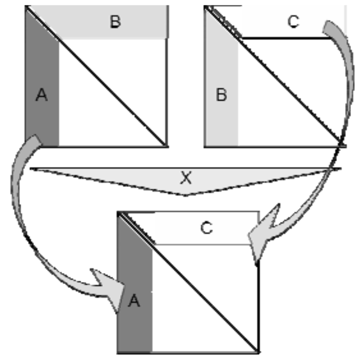

Figure 5. Combining the matrix pairs

The matrix combination can be determined according to the scheme shown in figure 6. In a MDSM with n domains, there are (n-1) pairs selectable if an intra-domain matrix (DSM-type) is being calculated (shown at the right side of figure 6). If an inter-domain matrix (DMM-type, left matrix of figure 6) is determined, (n-2) pairs are selectable. In the third step of the AID-Process, the indirect relations are calculated by multiplying the selected matrix pair. The resulting matrix contains the number of indirect relations for each element combination of the matrix pair. This is of course only possible if matrices are represented in a mathematically appropriate formulation, and contain only cyphers as relation marks. It is imaginable that a matrix multiplication is also done if the relation marks are real numbers representing the strength of the connection between the elements. However, weighted relations will be examined closer in future work. The number of indirect relations is the number of possible ways to connect the two elements by passing a third element (like element 2 is inserted between elements 1 and 3 in figure 2). The total number of indirect relations is added up for all matrix pairs (domains). For example, there may be five indirect relations between two specific components, due to three linkages through the resources- and two linkages through the functions-domain.

	A	B	C	D	E	F	G
A				(1)b			
B	(1)a		(2)a	X	(3)a	(...)a	(n-2)a
C				(2)b			
D							
E				(3)b			
F				(...)b			
G				(n-2)b			

	A	B	C	D	E	...	n
A		(1)b					
B	(1)a	X	(2)a	(3)a	(4)a	(...)a	(n-1)a
C		(2)b					
D		(3)b					
E		(4)b					
...		(...)b					
n		(n-1)b					

Figure 6. Selection scheme for the AID-Process

With matrix A having N rows and K columns, and matrix B having K rows and M columns, the indirect relations can be calculated using equation 1.

$$\mathbf{IR}(\mathbf{i},\mathbf{j}) = \sum_{j=1}^{M} \{\mathbf{A}(\mathbf{i},\mathbf{j}) \cdot \mathbf{B}(\mathbf{j},\mathbf{i})\},\ i = 1,\ldots,N \tag{1}$$

The resulting matrix IR is a new matrix with the number of indirect relations between the elements of the sub-matrices deriving from indirect relations in matrices A and B. If for example the components-functions and the functions-resources matrices are multiplied, the resulting matrix is a components-resources matrix that contains the number of indirect relations for each element combination. If there is more than one way to connect a component and a resource using a function, there may be more than one indirect relation indicated in the matrix field.

The process is repeated if more matrix pairs are to be calculated (see the selection scheme in figure 6). To get the total number of indirect relations, all resulting matrices are added up. In a MDSM with N domains, the total number of indirect relation can be calculated using equation 2.

$$\mathbf{I} = \sum_{i=1}^{N} \{\mathbf{IR}_i\} \tag{2}$$

We do not distinguish between different origins of indirect relations. For example, there is no difference between an indirect relation caused through a common component and an indirect relation caused through a common parameter setting. We plan to extend the AID algorithm to weight different origins of indirect relations to better identify critical relations in the matrix.

We identified three fundamentally different output scenarios from applying the AID-process. There may be a multitude of indirect relations or only few indirect relations; unusual but possible are no indirect relations between the domain elements.

If there are proportionally many indirect relations between two specific elements, there are many different ways to establish an indirect connection (i.e. a connection through another element) between them. In this case, a designer may also identify a direct relation between these elements. However, there may or may not be a direct relation “in reality”, depending on the context of the elements. If two components possess many indirect relations (through parameters, functions, resources, or tasks), the two elements are likely to be “somehow” related to each other. The designer has to decide if there is really a direct relation - and the AID-process calls his attention to the important cells of the matrix.

Proportionally few indirect relations indicate that there are only few ways to connect two elements. However, in a matrix of average density, the dense mesh of relations between elements causes a large quantity of indirect relations. In fact, nearly every element is connected through at least some indirect relations. That is why the case of few indirect relations between elements has little saying for MDSM analysis.

Some elements of the matrix may not be related indirectly at all, i.e. there is no way to connect these elements using a third one. The designer’s attention has to be called to this point, because he should look carefully if really no indirect relations exist.

The benefit of the AID algorithm is that designers receive the opportunity to identify critical relations within the matrix. These relations are characterized through either many (higher-than-average) or no indirect relations. Especially in large MDSMs, it is essential to support the designer during the matrix filling process and during MDSM analysis by highlighting important elements and reducing the complexity.

3.3 Visualization of the results

We suggest not to insert the explicit number of indirect relations in the MDSM. Instead, we propose a graphical visualization using four-fold color shading of the matrix cells to group different quantities of indirect relations. The following thresholds relative to the maximum quantity of indirect relations have turned out to be useful to distinct the three MDSM analysis scenarios:

- Darkest shading: [100%-80%]
- Medium shading:]80%-40%]
- Light shading:]40%-0%[

- White: - no indirect relation -

An exemplary matrix with shading (DMM type) is depicted in figure 7. The dark fields in the matrix therefore contain many indirect relations, i.e. there are many ways to establish an indirect connection between both elements through other element dependencies. If the dark-shaded fields also contain a mark (here: “1”), a direct relation between the elements exists.

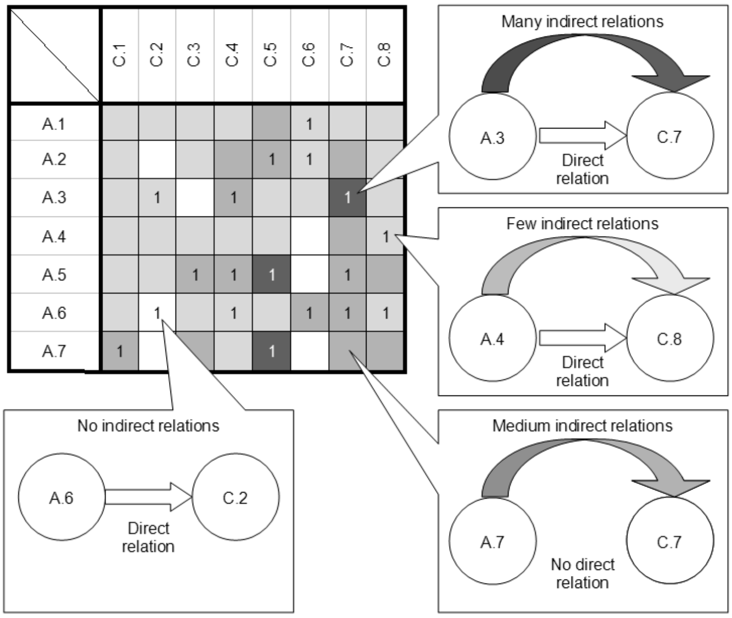

Figure 7. Example matrix with visualized indirect relations

The color shading helps the designer to identify important fields within the MDSM. Important fields are those containing many indirect relations. As already mentioned, the matrix filling process is time-consuming. AID can be used during a matrix filling process to speed up the process. If AID is used in this way, it can help designers to fill the matrix more quickly and more reliably, because important interdependencies are highlighted. AID may also be used as an analysis tool for filled matrices, as it depicts the relations that are important due to cross-domain interrelations. In addition, possible failures in a matrix can become apparent. There may for example be fields with many indirect relations (dark-shaded fields), but no direct relation. In this case, a designer should carefully look at these elements, as they may be directly related. There may also be fields with no indirect, but only a direct relation. These relations are isolated and do not belong to a block of elements. However, they may connect two blocks.

4. Conclusions and Further Work

The AID optimization process has been applied to multiple real-world examples to test the robustness and usability of the algorithm. Matrix sizes varied from 10 to 50 elements. We found that the quantity of indirect relations depends strongly on the density of the matrix. That is why it is important to use dynamic thresholds to identify fields with many indirect relations. The main advantage of the AID-algorithm is the quick visualization of important matrix areas and element relations. Using this visualization during a matrix filling process showed that indirect dependencies can be made clear to the team members more easily. However, to prove the assumption that many indirect relations are likely to cause also a direct relation between elements, more examples are to be conducted. The next research steps will therefore focus on the application of MDSM methodology and the use of AID as a filling aid for MDSMs. In addition, we plan to combine known matrix analysis methods and the AID algorithm to get further insight in the structure of multi-domain complex networks.

It is planned to integrate the MDSM methodology into the MOFLEPS [Maurer et al. 2005] analysis tool for complex network structures. This tool integrates several methods for complex network analysis and DSM analysis like clustering, partitioning or loop identification. By integrating the MDSM methodology into the tool, we will be able to examine products and product development processes under a holistic perspective.

References

Baldwin, C., Clark, K., "Design Rules - The Power of Modularity", MIT Press Cambridge, Mass., 2000.

Bongulielmi, L., Henseler, O., Puls, C., Meier, M., "The K&V-Matrix Method - An Approach in Analysis and Description of Variant Products", Proceedings of the International Conference on Engineering Design (ICED 01), Glasgow, 2001.

Browning, T., "Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions", IEEE Transactions on Engineering Management, Vol.48, No.3, 2001, pp 292-306.

Danilovic, M., Browning, T., "A Formal Approach for Domain Mapping Matrices (DMM) to Complement Design Structure Matrices (DSM)", Proceedings of the 6th Cambridge DSM Workshop, Cambridge, UK, 2004.

Hauser, J.; Clausing, D., "The House of Quality", Harvard Business Review, Vol.66, No.3, 1988, pp 63-73.

Kusiak, A., "Engineering Design – Products, Processes and Systems", Academic Press San Diego, 1999.

Malmqvist, J., "A Classification of Matrix-Based Methods for Product Modeling", Proceedings of Design 2002, 2002, Cavtat-Dubrovnik, Croatia.

Maurer, M., Boesch, N.-O., Sheng, G., Tzonev, B., "A Tool for Modelling Flexible Product Structures - MOFLEPS", Proceedings of the 15th International Conference on Engineering Design (ICED'05), Samuel, S., Lewis, W. (Ed.), Institution of Engineers, Melbourne, 2005, (CD-ROM).

Steward, D., "The Design Structure System: A Method for Managing the Design of Complex Systems", IEEE

Suh, N., "Axiomatic Design", Oxford University Press Oxford, 2001.

Transactions on Engineering Management, Vol.28, No.3, 1981, pp 79-83.

Maik Maurer
Technische Universitaet Muenchen, Institute of Product Development
Boltzmannstraße 15, 85748 Garching, Germany
Tel.: +49 89 28915155
Fax.: +49 89 28915144
Email: maik.maurer@pe.mw.tum.de
URL: http:// www.pe.mw.tum.de