9TH INTERNATIONAL DESIGN STRUCTURE MATRIX CONFERENCE, DSM'07
16 – 18 OCTOBER 2007, MUNICH, GERMANY

MODULE AND INTERFACE IDENTIFICATION AND DEFINITION – A COMPREHENSIVE APPROACH USING DSM

Robert Helmer[1], Ali Yassine[1] and Christoph Meier[2]
[1]Department of Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, USA
[2]Institute of Astronautics, Technische Universität München, Germany

Keywords: Product Development, Product Architecture, DSM Clustering, Module and Interface Definition

1 INTRODUCTION

Modularity has been widely recognized as a means to manage complexity in product design processes and achieve competitive advantages, mostly related to "business performance" [6]. For instance, reduction of development and assembly time, increased product variety and mass customization, reuse of modules in different products and ease of maintenance. Several DSM clustering algorithms have been presented in previous literature to automate the process of module identification, for example [5] and [7]. Still a high degree of arbitrariness exists due to the utilization of different rating schemes and the focus on different types of dependencies. While the distinction between different dependency types in DSM models ([2],[4]) increases accuracy of the model, how to actually handle these in DSM clustering has been rarely discussed, thus making it difficult for practitioners to apply these techniques successfully. Another issue that has hardly been investigated is the topic of overlapping clusters, minibuses and buses [3]. These structures constitute both design challenges and opportunities. We present a solution to these problems based on a new rating scheme, which is a slightly modified version of the one introduced by Pimmler et al. [2] and focuses on the *spatial* aspect for module identification. This modification, in addition to a case differentiation between different dependency types, allows handling different types of dependencies without the problem of choosing general weights as in a weighted sum approach or obtaining lots of different perspectives on the product architecture using multi-objective optimization techniques. Moreover, this modified rating scale and the case differentiation are the basis for a post-processing to correct and refine the clustering results. We propose different techniques to analyze and resolve possible overlappings, buses and minibuses, thus increasing modularity of the underlying product concept. Among these techniques is the possibility to use constraints in the clustering. This provides more accuracy in the definition of modules and their interfaces. Throughout the procedure we focus on "*assembly modules*", i.e. modules that can actually be built and therefore are decisive for the definition of interfaces, as opposed to "*design dependency modules*", for instance, signal modules which result from clustering signal design dependencies only, but usually constitute infeasible solutions.

2 PROPOSED APPROACH

Different rating scales have been presented in recent clustering algorithm publications. Although Pimmler et al. [2] introduced a rating that clearly maps each dependency weight to a distinguishable statement along with different types of interactions, which increases both accuracy and information content in the DSM model, this rating scale has not been adopted in automated clustering techniques. The rating scheme was refined later on by Sosa et al. [4]. Nevertheless, recent clustering publications did not make use of the increased amount of information in the model. Rather they used either single dependency types only or blended several aspects in one rating. Yet, poor data quality decreases the benefits gained from the optimization and is a source for unnecessary design iterations. Typically, one is tempted to tackle multiple types of interactions with either multi-objective optimization algorithms or by reducing the problem to a single-objective, for example, by use of a weighted sum approach. Both methods have advantages and disadvantages. Single-objective techniques allow "choosing" a certain product perspective and using an available single-objective clustering algorithm, while having

the drawback that it is all but trivial to find sensible weights. Furthermore, it is not sensible to give certain dependency types generally a higher or lower importance than others in clustering. Multi-objective clustering, on the other hand, provides lots of different perspectives of the product in one single optimization run, each optimal in its own merit (multi-dimensional Pareto-front of optimal solutions), without the need to specify weights. However, even if techniques to reduce the number of clustering results can be conceived, it is very difficult to compare various DSMs of the same complex product. A problem of both techniques in combination with the existing rating scale ([2],[4]) is that all required interactions of the same type are weighted equally for clustering algorithms.

2.1 New Rating Scale

Rather than considering *spatial* as an extra type of dependency, we see the *spatial* aspect more as the result of other dependency types, namely structural/mechanical, energy, signal and material interactions. The *spatial* aspect is decisive for the assignment of elements to modules. Elements at different ends of the product can usually not be in the same module, unless connecting elements are also assigned to this module. Therefore, the original rating scheme was modified as shown in Figure 1. The most important difference is the distinction between +1 and +2.

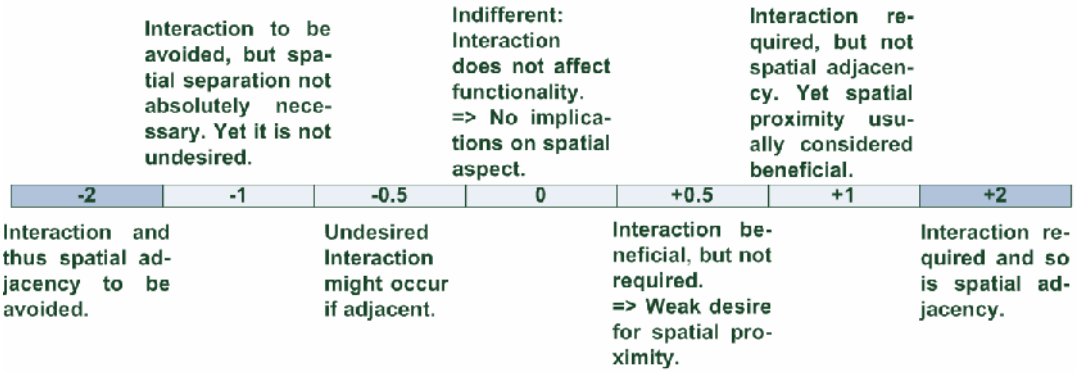


Figure 1: New Rating Scale

An advantage of this rating scale is that not only the necessity of an interaction is addressed, but also the possible requirement for spatial adjacency. For example, a required material exchange alone does not tell if this means two components have to be adjacent, or if they might also be situated at different ends of the product. For instance, the fuel tank of a car in the end delivers to the motor assembly (→ required flow), but obviously this does not demand spatial adjacency. So the pure flow requirement alone is not sufficient information. With the original rating scale this information was not captured.

2.2 Perspective Reduction

By applying the rating scale above independently to each of the four types of interactions mentioned above, the DSM has four entries per cell. Each entry tells something about the spatial requirement due to a certain type of dependency. Now it is possible to compare these different entries reducing them to a single value for the *overall* spatial requirement of the interactions. As an example, imagine a negative energy exchange between two elements if the elements are close, which, however, would not mean the components must not be adjacent (-1) and a required material exchange between the same elements requiring spatial adjacency (+2). It is obvious, that the requirement for spatial adjacency prevails in order to achieve proper functionality, while the negative interaction has to be accepted in this product concept. The reduction of all interactions to one value representing the overall spatial demand of all dependencies is advantageous in several ways for the further use of the DSM model in clustering and refinement of the product architecture: (1) it allows using available single-objective clustering techniques; (2) there is no bias towards any of the interaction types; a case differentiation is performed in every single cell, so that any aspect can prevail; (3) it eases review of the clustering result, because one can primarily concentrate on this single value and there is only one clustering result per optimization run; (4) the reduction allows for the classification of different types of overlapping clusters, minibuses and buses, thus giving insights that can be used to further improve the result of the clustering (see post-processing).

2.3 DSM Clustering and Post-Processing

Yu et al. [7] recently developed a clustering algorithm based on the Model Description Length (MDL) principle. The MDL-based objective function exactly captures the essential task of the clustering. In combination with a robust genetic algorithm (GA), this algorithm has been proven capable of handling DSMs of complex products and teams reliably. Some modifications have been made to improve the GA, e.g. to handle both positive and negative dependencies and to prevent elements from being members of more than two clusters. However, details of this modification are not addressed here. Rather we focus on the investigation of overlappings, minibuses and buses. While it is generally agreed upon that these constitute areas requiring increased system engineering attention, opportunities and maybe risks have not been discussed widely. A couple of ways to deal with overlapping clusters in the DSM model have been named by Hölttä [1]: (1) merge clusters, (2) assign elements to one of the clusters, (3) duplicate the element and assign one to each cluster, (4) leave overlapping as is.

The treatment of overlapping clusters poses the question why no such ways have been presented and discussed for minibuses and buses, because an overlapping cluster element is nothing but a special minibus that has most dependencies with elements of two clusters instead of three or more.

Therefore, we extend the list above by two new notions: *cloning* and *splitting*. Cloning is in fact similar to duplication, with the only difference being that duplication means to create two such elements, while cloning denotes the multiplication without limits. Splitting looks similar to duplication and cloning on first glance, but in fact differs from these options in that it suggests splitting of distributed elements into *coupled* subelements. The definition of interfaces between these subelements can often be done in an additional spatially constrained DSM, which we call Second Level DSM as opposed to the First Level DSM, which is the DSM comprising the entire product.

3 CONCLUSION

We present a comprehensive approach towards the definition of modules and their interfaces using DSM, comprising all aspects, from data acquisition over handling of multiple objectives in clustering and the clustering algorithm itself, to a semi-automated post-processing. This improves accuracy in this process and can potentially reduce design iterations and thus development time and cost. Furthermore, it aims at providing a consistent method for practitioners to make module identification and definition with DSM more a systematic technique than an art.

REFERENCES

[1] Hölttä, K. “Modular Product Platform Design”, Doctoral Dissertation, Helsinki University of Technology, Espoo, Finland, August 2005.

[2] Pimmler, T. and Eppinger, S. “Integration Analysis of Product Decompositions”, *ASME Design Theory and Methodology Conference*, Minneapolis, MN, September 1994.

[3] Sharman, D. and Yassine, A. “Characterizing Complex Product Architectures”, *Systems Engineering Journal*, Vol. 7, No. 1, 2004.

[4] Sosa, M., Eppinger, S., Rowles, C. “Designing Modular and Integrative Systems”. *ASME 2000 International Design Engineering Technical Conferences*, Baltimore, 2000.

[5] Whitfield, R.I., Smith, J.S., Duffy, A.H.B. “Identifying Component Modules”, *Seventh International Conference on Artificial Intelligence in Design (AID’02),* Cambridge, U.K., 2002.

[6] Whitney, D. “Physical Limits to Modularity”, *Working Paper, ESD-WP-2003-01.03-ESD*, Massachusetts Institute of Technology, Cambridge, MA, USA, 2003.

[7] Yu, T., Yassine, A., Goldberg, D. “An Information Theoretic Method for Developing Modular Architectures Using Genetic Algorithms”, *Research in Engineering Design,* Forthcoming, 2007.

Contact: Robert Helmer
Industrial and Enterprise Systems Engineering
University of Illinois at Urbana-Champaign
104 S. Mathews
Urbana, IL 61801
USA
Tel.: +01-217-333-7621
rhelmer@uiuc.edu
http://www.iese.uiuc.edu/pdlab/

9TH INTERNATIONAL DSM CONFERENCE

Module and Interface Identification and Definition

- A Comprehensive Approach Using DSM

Robert Helmer[1], Ali Yassine[1], Christoph Meier[2]

[1]Product Development Research Laboratory
Industrial and Enterprise Systems Engineering
University of Illinois at Urbana-Champaign
USA

[2]Institute of Astronautics
Technische Universität München
Germany

IN COOPERATION WITH BMW GROUP
CAPITALIZE ON COMPLEXITY

Index

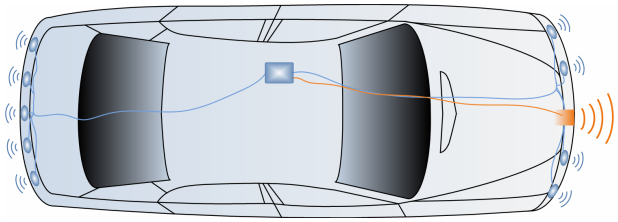
Benefits of Modularity

- Successful modular designs in both hardware and software products
- Large variety of benefits throughout the entire product life cycle, mostly related to “business performance”:
  - Parallel development and assembly
  - Product variety by different combinations of modules
  - Reuse of modules in different product generations
  - Ease of maintenance and recycling
  - Basis for product platform design
  - …

→ Successful modular design requires well-defined interfaces!

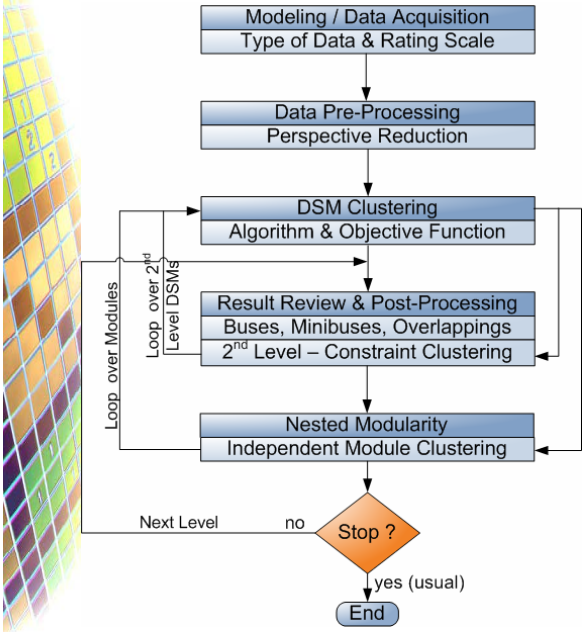
Spatial Aspect as Driving Factor for Module Definition

- For module definition at system level, we find that *spatial* aspect is driving factor
- *Spatial* constraints on the design
  → Maximizing interactions within and minimizing interactions between modules possible only to certain degree
  → Weak interactions can outweigh relatively strong interactions due to *spatial* requirements
- Examples:
  - Automobile driver assistant systems
  - Distributed Control Systems (DCS)



→ Almost all types of systems are subjected to similar *spatial* constraints

Concept for Definition of Modules and their Interfaces



Situation:

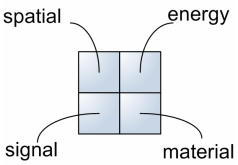
- Interface definition often dealt with insufficiently or late
- Highest leverage in cost savings in early phases of development process
- Wrong assumptions about interfaces cause major design iterations

Goals:

- Definition of modules & interfaces
  - Early in the development process
  - Accurately

Data Acquisition – Types of Data and Rating Scales

- Previously different types of data and different rating scales used to model systems:
  - Binary: 0 / 1
  - Weighted: e.g. 0….10 from no to strong interaction
  - Pimmler et al. [1]: 4 interaction types and mapping between rating scale and distinguishable statements



+2	Interaction necessary for functionality
+1	Interaction beneficial for functionality
0	Indifferent
-1	Interaction causes negative effects
-2	Interaction must be prevented

  - Sosa et al. [2]: add *structural* dependency; spatial aspect still captures adjacency due to alignment, orientation, serviceability, etc.

Problem: How to use information about different dependency types in module clustering?

DSM Module Clustering – a Multi-Objective Optimization Task (1)

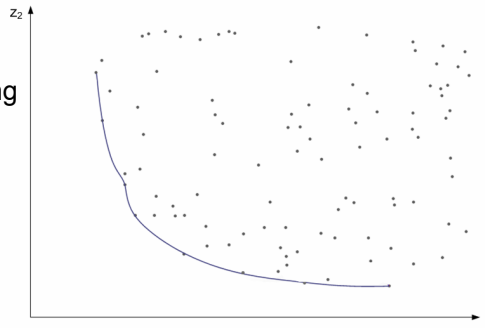
- Different dependency types ([1],[2])
  Problem: Usually clustering not optimal for all types at a time
  → Search for an optimal trade-off

- Possibilities to deal with multiple objectives in DSM clustering:
  - Cluster based on one type only
  - Reduction to single objective:
    - *Constraint approach*
    - *Weighted metric method*
    - *Weighted sum:* $f_{ges} = w_1 f_{obj1} + w_2 f_{obj2} + w_3 f_{obj3} + \ldots + w_n f_{objn}$

    ⊕ Use of single-objective optimization algorithms
    ⊖ Choice of weights, constraint values or virtual solution
    ⊖ Distortions with weighted sums

DSM Module Clustering – a Multi-Objective Optimization Task (2)

- Multi-Objective Optimization
  ⊕ No choice of weights before clustering
  ⊖ Lots of Pareto-optimal solutions
  ⊖ Comparison very difficult



Pareto-Front for 2 objectives

- Problem with all mentioned techniques and the existing rating scale ([1]):
  All required interactions of the same type are weighted equally
- But: some may require spatial adjacency, others don't

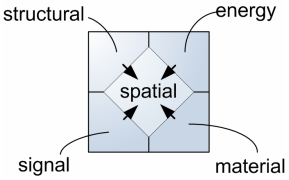
Pareto-Front for 3 objectives

Special Approach for DSM Module Clustering – New Rating Scale and *Perspective Reduction*

- Modification of Pimmler's [1] rating scale:

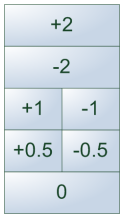
-2	-1	-0.5	0	+0.5	+1	+2

- Distinction between "*interaction and spatial adjacency required*" (+2) and "*interaction, but not spatial adjacency required*" (+1)



Perspective Reduction = case differentiation in each cell of the DSM resulting in one single value of the overall spatial requirement

- ⊕ No general bias towards any of the interaction types and no weights have to be chosen
- ⊕ Use of existing single-objective clustering algorithms possible, only one solution
- ⊕ No distortions



Dependency ranking for *Perspective Reduction*

Distortions with Weighted Sums

- Weighted sums distort clustering due to
  - Double counting of same interaction
  - Influence of marks in addition to +2 marks, for instance

General weighted sum for module clustering:

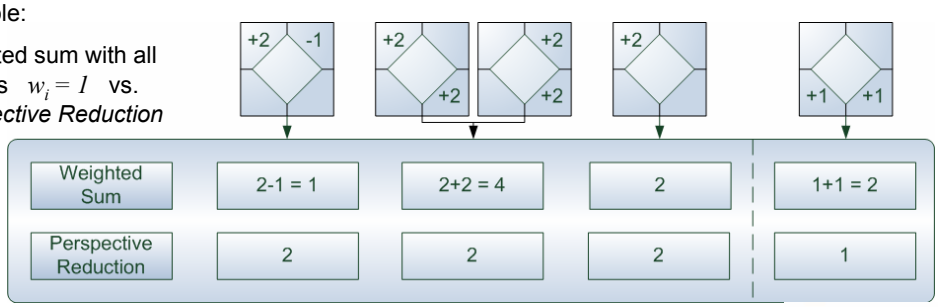
$$f_{ges} = w_1 f_{structural} + w_2 f_{energy} + w_3 f_{signal} + w_4 f_{material}$$

Dependency ranking for *Perspective Reduction*:

+2	-2	+1	+0.5	0
		-1	-0.5	

Example:

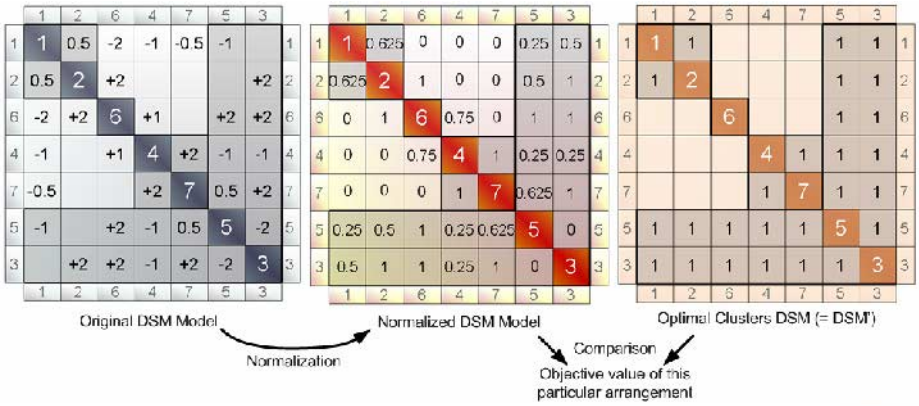
Weighted sum with all weights $w_i = 1$ vs. *Perspective Reduction*



Model Description Length Based Objective Function

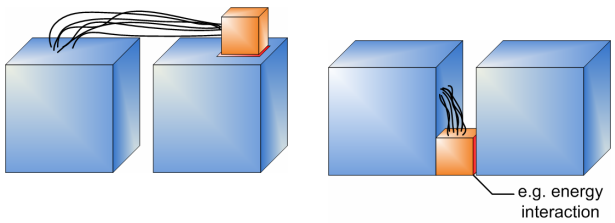
- Yu [4]: clustering objective function based on *Model Description Length (MDL)*
  → MDL = number of bits required to describe the DSM model
- Used Genetic Algorithm allows for overlapping of clusters and detection of buses
- MDL aims at finding best "macroscopic" clustering arrangement

$$f_{DSM} = (1-\alpha-\beta)\cdot\left(n_c + \sum_{i=1}^{n_c} s_i\right)\cdot\log_2 n_n + \alpha\cdot S_1\cdot(2\log_2 n_n + 1) + \beta\cdot S_2\cdot(2\log_2 n_n + 1)$$



Post-Processing

- Direct implementation of clustering results usually not sensible
- What to do with overlapping clusters, minibuses and buses?
- "…areas requiring integration across chunks…" ([1])
  → How can DSM be involved in this process?
- Impossible to tune clustering algorithms for all possible interaction mark constellations
  → Necessity to review the clustering result



E.g. "+2" spatial adjacency requirement does not necessarily mean the element has to be in the corresponding module.

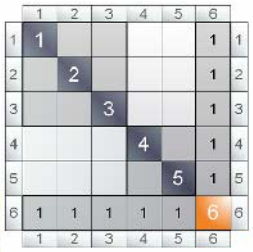
→ +2 may appear outside module bounds

→ manual review required

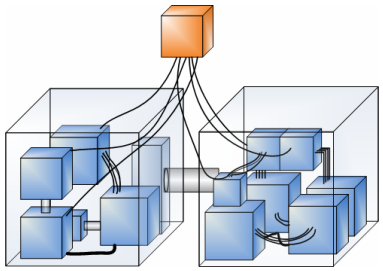
→ Post-processing = correction and refinement of the clustering result

- Semi-automated procedure
- Possible due to new rating scale

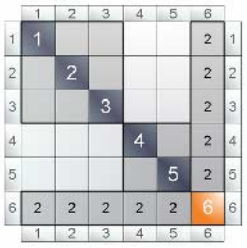
Post-Processing – Bus Types



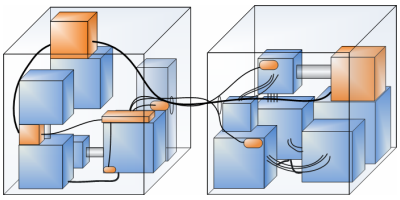
+1 Bus

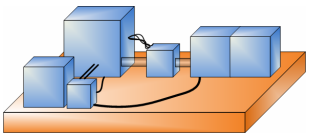


+1 Bus: interaction, but spatial adjacency not required



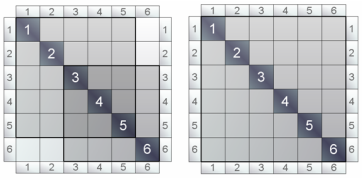
+2 Bus



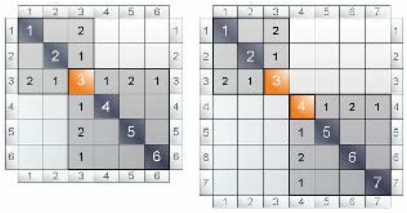


+2 Bus: interaction and spatial adjacency required

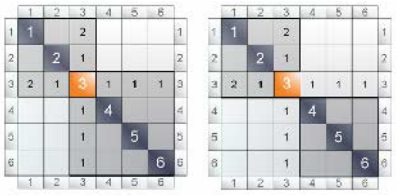
Post-Processing – Dealing with Buses, Minibuses & Overlappings

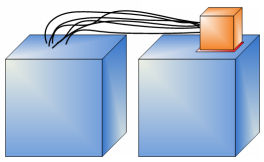


Merging: large overlapping → merged clusters

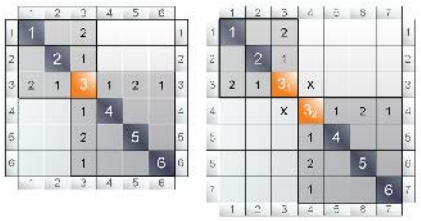


Cloning: one element → several identical elements

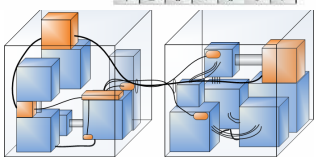




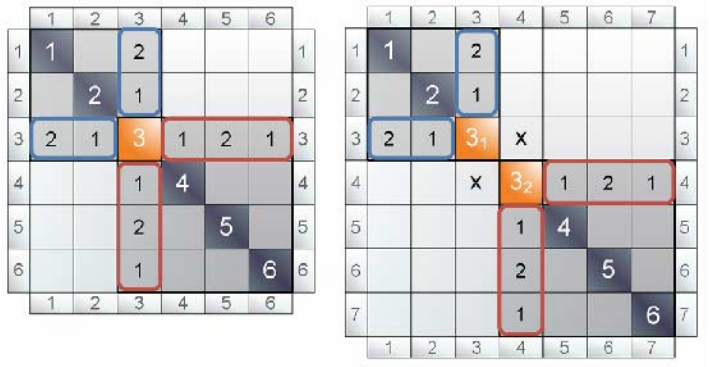
Assignment to one of the modules, usually complying with +2 spatial adjacency requirement.



Splitting: Split distributed system into two or more subsystems or divide structures in segments.



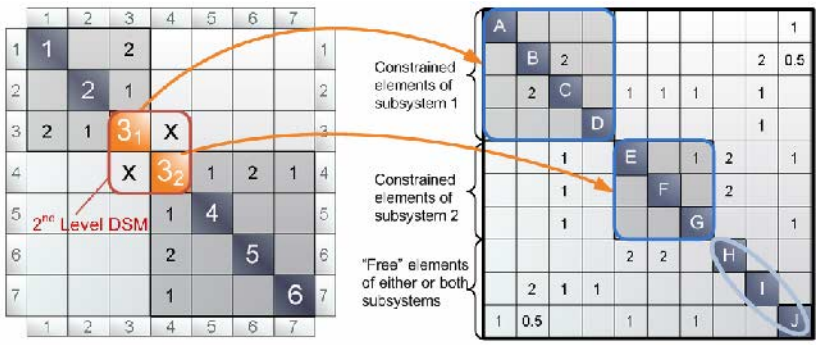
Post-Processing – Splitting (1)



- Distributed system split into subsystems, which are assigned to modules
- "x" denote any kind of interaction between the subsystems → to be defined
- Which element of the system is in which subsystem?

Post-Processing – Splitting (2)
Definition of Subsystem Elements & Interfaces

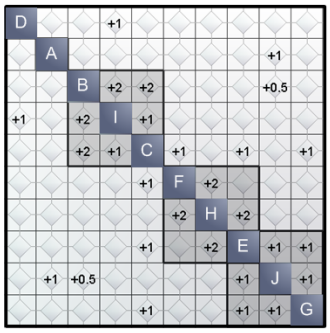
- Level of resolution for system under consideration is increased
- Separate 2nd Level DSM to show internal interactions between elements of this system
- +2 interactions (requirements for spatial adjacency) to elements outside the system considered via constraints on the 2nd Level DSM clustering
- Example:



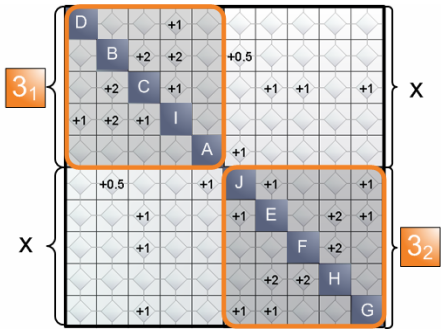
Split 1st Level DSM 2nd Level DSM

Post-Processing – Splitting (3)
Definition of Subsystem Elements & Interfaces

- Clustering the 2nd level DSM
  → Assignment of subsystem elements to modules



2nd Level DSM



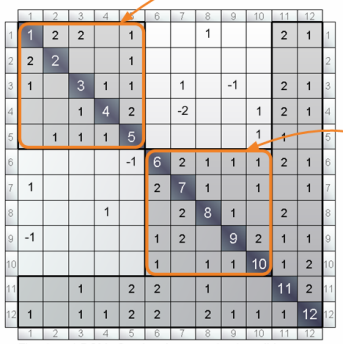
2nd Level DSM

Unconstrained clustering:
→ Wrong module definitions and thus wrong interfaces

Constrained clustering:
→ Only "free" (i.e. unconstrained) elements change cluster membership

Post-Processing – Investigation of Nested Modularity

- Modularity on different levels → modules inside modules
- Identification of decoupled entities → e.g. further benefits for assembly
- Independent clustering of 1st Level DSM modules
  → Overall MDL-based objective value stays constant



First Level DSM: Overlappings resolved or did not occur, buses left as they were, because expert decided that splitting or cloning not applicable or desired.

Cluster 1 consists of modular substructures

Cluster 1 clustered independently of rest of first level DSM

No modular structures could be identified for cluster 2

Cluster 2 clustered independently of rest of first level DSM

Conclusions and Future Work

Conclusions:

- New rating scale allows for active module and interface definition
- Constrained clustering for correct module definition
- Consistent method and increased accuracy
- Modularity not a feature inherent in a certain product concept, but can actively be influenced
- Danger of over-modularization

Future work:

- Future work could extend the clustering algorithm to take into account information about sampling rates, signal bandwidth, mass flows, etc.
- But: Further aspects have to be taken into account and model complexity increases

IN COOPERATION WITH BMW GROUP
CAPITALIZE ON COMPLEXITY

References

[1] Pimmler, T. and Eppinger, S. “Integration Analysis of Product Decompositions”, *ASME Design Theory and Methodology Conference*, Minneapolis, MN, 1994.

[2] Sosa, M., Eppinger, S., Rowles, C. “Designing Modular and Integrative Systems”. *ASME 2000 International Design Engineering Technical Conferences*, Baltimore, 2000.

[3] Yu, T.-L., Yassine, A., Goldberg, D. “An Information Theoretic Method for Developing Modular Architectures Using Genetic Algorithms”, *Research in Engineering Design,* Forthcoming, 2007.