INTERNATIONAL DESIGN CONFERENCE - DESIGN 2010
Dubrovnik - Croatia, May 17 - 20, 2010.

DESIGN 2010

A SYSTEMATIC QUALITATIVE COMPARISON OF FIVE APPROACHES TO MODULARITY

F. Borjesson

Keywords: modular function deployment, design structure matrix, heuristics, design for X, qualitative comparison

1. Introduction

The need to compare alternative approaches to modularity in a systematic way has arisen from the research idea that new hybrid approaches may be created to improve the main approaches on which they build. Ulrich and Eppinger [Ulrich, Eppinger, 2008] start with a simple idea, the chunk. Chunks are physical building blocks. Modular architecture has the following properties: (a) chunks implement one or a few functional elements in their entirety, and (b) the interactions between chunks are well defined and are generally fundamental to the primary functions of the product. We add one more point from [Erixon 1998]: A module is a physical building block with standardized interfaces selected for company-specific reasons. In the remainder of this paper, the phrase "approach to modularity" will be used to mean methods by which modular architectures are defined. We will look at five such approaches. According to [Hölttä-Otto, 2005] there are three main approaches to modularity: (1) Heuristics, (2) Design Structure Matrix (DSM), and (3) Modular Function Deployment (MFD). In addition to these, we will look at two hybrid approaches: (4) Functional-Strategic DSM [Blackenfelt, 2001], and (5) Extended Implementation Structure Matrix [Sellgren, Andersson, 2005].
The purpose of the present paper is to answer the following two questions: (1) How may we compare the approaches in a consistent manner? (2) Do derived approaches improve on the main approaches they build on?

2. Overview of Methods

For the purpose of providing a brief overview, we will focus on five critical aspects of each method: how data is organized, data types that can be represented, relationships captured, type of interactions or dependencies, and how modules are generated.
Organization of data is particularly important in large projects and for selecting the right type of computer software.
Data type refers to one or several of the following: Technical Solutions (abbrev. TS; to describe how functions are realized), Customer Requirements (abbrev. CR; to describe the benefits customers are looking for in the product), Product Properties (abbrev. PP; to be able to make quantitative statements about the performance level of certain functions), Module Drivers (abbrev. MD; to describe the company specific strategy), Functional Requirements (abbrev. FR; statements about functions that must be performed by the product, used primarily in synthesis), and Functions (abbrev. FU; transformation of an input into an output, often expressed as verb plus noun, used primarily in analysis).
Relation is a reference to comparisons between pairs of data types. For example, QFD is a relation between Customer Requirements and Product Properties.

Interaction is the aspect of the Relation we are interested in. In a Task-based DSM, for example, the interaction is related to Design sequence, e.g., which Technical Solution of the two would be designed first. Other Interactions are Degree of causality (to what extent does changing one drive change in the other), Spatial (should Technical Solutions be close or far apart in the final product), and Flow (exchange of Energy, Matter, or Information).
Module generation refers to definition of groups of TSs into Modules. The Methods take three approaches to Module generation: Hierarchical Clustering creates groups where the TSs have similar PPs and MDs, Least interaction minimizes the dependency between groups of TSs, and Rule-based looks for certain patterns of Flow.

		Heur.	DSM	eISM	MFD	FS-DSM
Organi-zation of data	Matrix		✓	✓	✓	✓
	Func. Struct. diagram	✓				
Data types	Technical Sol'ns (TS)	✓	✓	✓	✓	✓
	Customer Req'ts (CR)				✓	
	Product Properties (PP)				✓	
	Module Drivers (MD)				✓	✓
	Functional Req'ts (FR)			✓		
	Functions (FU)	✓				
Rela-tions	TS-TS		✓	✓		✓
	TS-MD				✓	✓
	TS-PP				✓	
	CR-PP				✓	
	CR-FR			✓		
	TS-FR			✓		
	TS-FU	✓				
	FU-FU	✓				
Inter-action	Spatial		✓	✓		✓
	Degree of causality			✓	✓	
	Design sequence		✓			
	Flow (Energy / Matter / Info.)	✓	✓			✓
Module genera-tion	Hierarchical Clustering				✓	
	Least interaction		✓	✓		
	Rule-based	✓				✓

Figure 1. Overview of all five methods

In Figure 1, a black checkmark means the line item in question is applicable to the particular Method; a shaded checkmark means normally it is not, but with a minor modification it could be. DSM is a reference to Task-based and Component-based DSMs, see section 2.2.

2.1 Heuristics

Within modular architecture, heuristics try to capture how designers actually think. According to [Gilovich, Griffin, Kahneman 2002], heuristics are based on patterns of biased judgments, represent sensible estimation procedures, yield "quick and dirty" solutions, draw on underlying processes that are highly sophisticated, and are normal intuitive responses to even the simplest questions about likelihood, frequency, and prediction. The heuristics we look at here are based on flow of matter, energy,

and information between functional elements in a function-structure diagram [Stone, Wood, Crawford 1998]. Table 1 summarizes the three rules. Readers interested in other heuristics (with some features similar to Module Drivers) may refer to [Zamirowski, Otto 1999].

Table 1. Heuristics related to flow [Stone, Wood, Crawford 1998]

Heuristic	Description
Dominant flow	If the same flow of matter, energy, or information goes through a sequence of functions, they should form a module.
Branching flow	If a flow splits up into parallel function chains, the subfunctions that make up those chains should form modules.
Conversion-transmission	Functions that convert one type of flow into another should form modules. If the conversion is followed by transmission, that should be part of the same module.

[Hölttä-Otto, 2005] compares all main approaches on their level of repeatability and offers a score based on the ratio of students that successfully apply each of the approaches. The heuristics in Table 1 scored quite high, in particular the application of Conversion-transmission. However, repeatability of a given flow heuristic might be high on a given function structure diagram, but in general the creation of the diagram itself is not a highly repeatable activity. This is supported by [Ulrich, Eppinger 2008] saying "There is no single correct way of creating a function diagram and no single correct functional decomposition of a product." In contrast, [Kurfman et al 2003] achieved 80% repeatability in an experiment where groups of subjects analyzed a toy ball gun with 15-20 functions to create a functional model, using a particular method. In the author's experience, repeatability would be lower for significantly more complex products.

2.2 Design Structure Matrix (DSM)

DSM may be thought of as a generic way of mapping interdependencies. Component-based DSM can be used to define modules in a product architecture [Hölttä-Otto 2005]. Task-based DSM may be used to determine the ideal sequence of development tasks in a project [Ulrich, Eppinger 2008]. The Component-based DSM in Figure 2 shows the task of developing B can only be completed once the task of developing A is complete: these are sequential. The tasks of developing C and D both depend on the task of developing B, but once it is concluded, C and D can be developed in parallel. Finally, the tasks of developing E and F are coupled. The best sequence is one that minimizes the number of coupled tasks. DSM predicts E and F should form a module [ibid.].

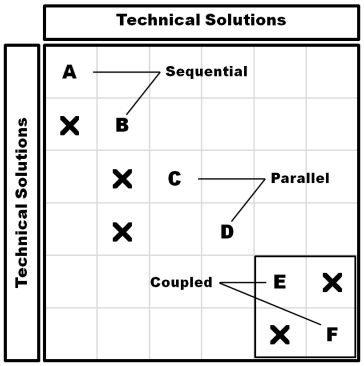

Figure 2. DSM is based on mapping dependencies

2.3 Modular Function Deployment (MFD)

MFD [Erixon, 1998] is based on the idea of decomposing CRs into specific statements and linking them to measurable and controllable PPs, decomposing the product into TSs, describing how each TS impacts the performance on a particular PP, and grouping TSs carrying similar properties and strategic intent to define modules. Figure 3 shows how CRs, PPs, TSs, and MDs are visualized in MFD.

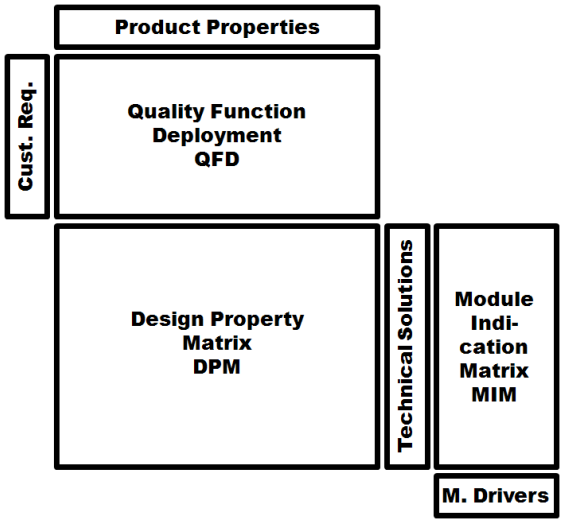

Figure 3. MFD uses three interlinked matrices

The grouping of Technical solutions by Product Property and Module Driver may be done manually or using statistical methods such as hierarchical clustering. Module Drivers are central and unique to MFD, and are presented below in Table 2.

Table 2. The Module Drivers

Module Driver	Strategy	Module Driver	Strategy
Common unit	Use solutions in many variants	Recycling	Simplify scrapping
Carry over	Use solutions in future generations	Strategic supplier available	Use external partner to develop, produce etc
Technical specification	Change specification level	Separate testability	Test separately before final assembly
Styling	Create styling variation	Upgrading	To increase after-sales
Planned design change	Allow for design changes	Process / organization	Protect scare resources in production or design
Technology push	Incorporate new technology	Service / Maintenance	Easy field replacement

2.4 Functional-Strategic DSM

This method [Blackenfelt, 2001] is a hybrid between DSM, MFD, and Heuristics. From DSM, it takes the format for describing dependencies. From MFD, it adds strategic considerations, but using the Condensed module drivers in Table 3, instead of the original twelve (Table 2). From Heuristics, it adds flow of Matter, Information, and Energy. To this, Blackenfelt adds degree of Spatial interaction.

Table 3. Condensed module drivers [Blackenfelt 2001]

Condensed module drivers	Original twelve Module Drivers	
Commonality	Technical Specification, Styling	Common Unit
Carry Over	Technology Push, Planned Development	Carry Over
Make or Buy	Process/Organization	Strategic Supplier
Life Cycle		Separate Testability, Service/Maintenance, Upgrading, Recycling

Some Module Drivers are mutually conflicting. As an example, a conflict exists between Technical Specification (several performance levels) and Common Unit (one level only). Module Drivers belonging to the same Condensed module driver are said to be supporting if they appear on the same side of the dotted line in Table 3, conflicting otherwise.

For any pair of Technical Solutions, conflicting or supporting strategic objectives are indicated using a scale from -2 to +2 in the Strategic DSM. A similar scheme is used in the Functional DSM. For example, a score of +2 on Spatial would imply two Technical Solutions must be adjacent in space to function; a -2 would signify they absolutely may not be. Figure 4 shows the template for the two matrices. CO, C, MB, and LC refer to the Condensed drivers. S, M, I, E refer to Spatial, Matter, Information, and Energy, respectively.

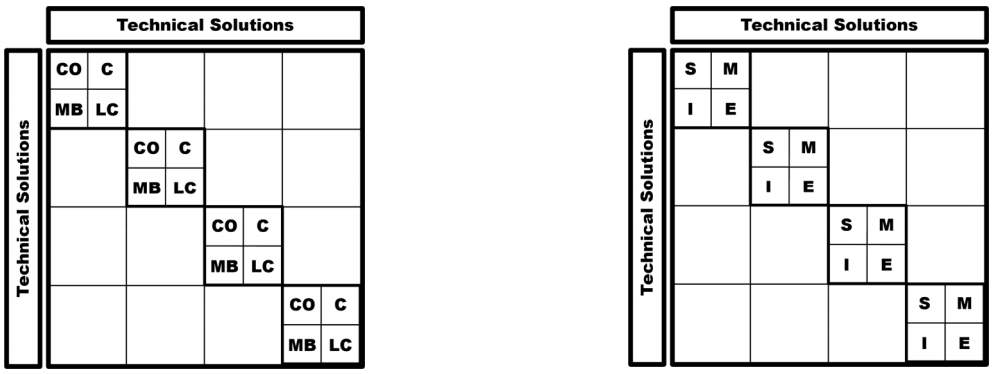

Figure 4. Strategic DSM (left) and Functional DSM (right)

The process of generating modules involves a three-step rule-based algorithm operating on both matrices.

2.5 Extended Implementation Structure Matrix (eISM)

eISM [Sellgren, Andersson 2005] is a hybrid of DSM and MFD. From DSM, it takes the format for describing dependencies. From MFD, it takes QFD and DPM, with FRs replacing PPs. Like MFD, the eISM uses three interlinked matrices, as shown in Figure 5.

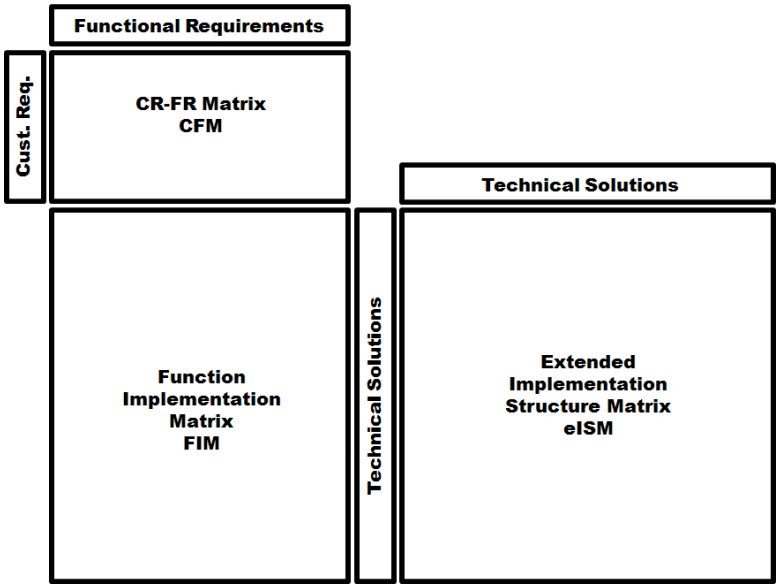

Figure 5. eISM uses three interlinked matrices

The creators of eISM, state the purpose of their approach is “to find a way to bridge the gap between the ‘hard’ technical requirements and the more ‘soft’ interactive requirements” [ibid.]. The FIM represents that bridge. Functional Requirements are stated as verb/noun combinations, which allows eISM to describe more easily how the product is used. A comparison of Figure 3 and Figure 5 shows both MFD and eISM translate CRs into TSs using an intermediate data type. The advantage of PPs is they can be measured, controlled, and assigned a goal value. This is not possible for FRs which may

be a disadvantage in many practical applications. On the other hand, soft interactive requirements are harder to describe with PPs. In the end, the choice of PP or FR would depend on the application.

3. Method of comparison

3.1 Initial set of criteria for comparison

The criteria used in this comparison are based on the works by [Huang, 1996], [Hölttä-Otto, 2005], [Keller, Binz, 2009] and own project experience. The criteria are shown in summary format below.

Table 4. Summary of criteria used in analysis

Source	Category	Criteria
Huang	Functionality requirements	Gather and present facts, Measure performance, Evaluate whether design is good enough, Compare design alternatives, Highlight strengths and weaknesses, Diagnose why an area is strong or weak, Provide redesign advice, Predict what-if effects, Carry out improvements, Allow iterations to take place
	Operability requirements	Easy to learn or well-known concepts, Systematic (all relevant issues considered), Represent product and process data, Teaches good practice, Little effort for designer, Implementation cost and effort, Rapidly effective, Stimulates creativity
	Flexibility & Focus	Allows some degree of flexibility, Reasonably accurate
Hölttä-Otto	Overall requirements	Identifies commonality between products in family, Identifies interfaces that are simple, Approach is easy to use, Module output is repeatable, Module output is feasible (realistic)
Keller/ Binz	Revisability	Validation, Verification
	Pract. Relev. & Competitiveness	Innovativeness, Competitiveness
	Scientific Sound.	Objectivity, Reliability, Validity
	Comprehensibility	Comprehensibility, Repeatability, Learnability, Applicability
	Usefulness	Effectivity, Efficiency
	Prob. Specificity	Problem Specificity
	Struct. & Compatibility	Handling Complexity, Problem Solving Cycle, Structuring, Compatibility
	Flexibility	Flexibility
Experi-ence	Overall requirements	Describes customer requirements, Allows for concurrency, Features integrated views of data, Accounts for basic physics, Captures product geometry, Supports strategic objectives, Can use software support, Simplify handover to design, Allows adjustment to tool itself

Huang's requirements are very focused on what-if-scenario modeling and ease of use. The Hölttä-Otto requirements are very much geared toward interface generation and usefulness of output. The Keller/Binz requirements are very comprehensive but not necessarily specific to architecture generation. The author's own requirements are architecture-specific and somewhat similar to the Hölttä-Otto requirements, but neither set is as comprehensive as either Huang or Keller/Binz. For these reasons, it was relevant to find one set of criteria taken from these models.

3.2 Grouping the criteria

A dendrogram shows the relative proximity of the different criteria. To create a dendrogram using Ward's method, a table of distances was established by pairwise comparison.

4. Results

The resulting dendrogram is shown in Figure 6. Where the dendrogram crosses the gray line, there are 11 subclusters. The final list includes a bias toward the criteria based on Experience, since several of those were not adequately captured in the group of 11 subclusters.

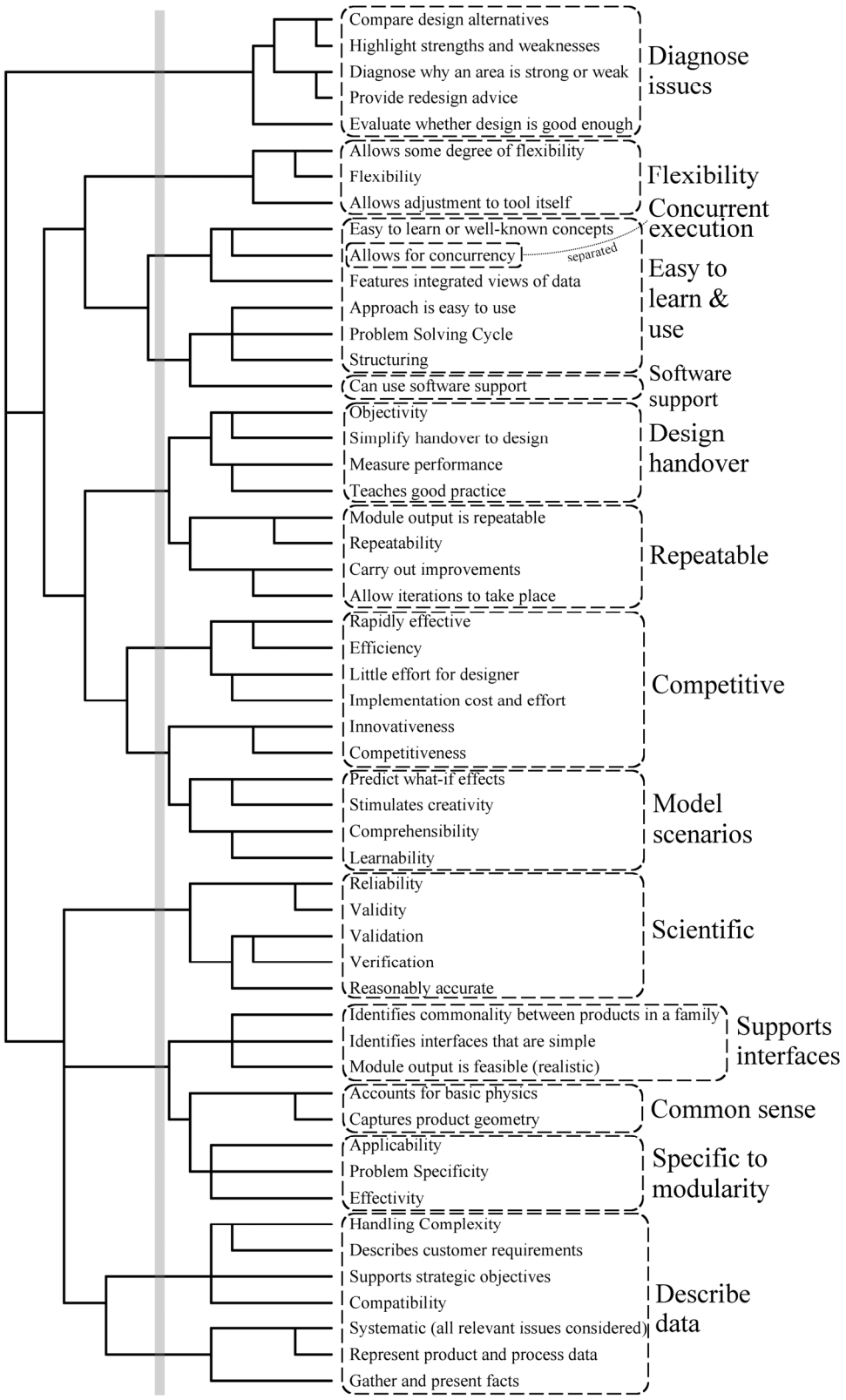

Figure 6. Dendrogram of the criteria in Table 4

Figure 6 indicates that both "Diagnose issues" and "Model scenarios" are potential evaluation criteria. However, it was found that none of the methods really score on either criteria. Therefore, those two were excluded. Also, "Allows for concurrency" does not seem to belong in "Easy to learn & use" and was therefore separated and included as "Concurrent execution". The result is shown in Table 5.

Table 5. Final list of 12 criteria

Criteria	Explanation
Flexibility	Method is flexible, allows adjustments
Concurrent execution	Promotes concurrent execution in groups
Easy to learn & use	Easy to learn, well-known concepts
Software support	Conducive to software support, including large projects
Design handover	Simplify handover from concept phase to detailed design
Repeatable	Method is repeatable and allows iterations
Competitive	Method represents an improvement over existing methods
Scientific	Based on science, valid, verifiable, accurate
Support interfaces	Supports generation of interfaces in modular architecture
Common sense	Allow common sense (physics & product geometry)
Specific to modularity	Specific to generation of modular architecture
Describe data	All data, including customer requirements and strategic intent

To determine whether the criteria seem relevant, the author made an experience-based assessment of each approach, using the 12 criteria. The result is shown in Figure 7.

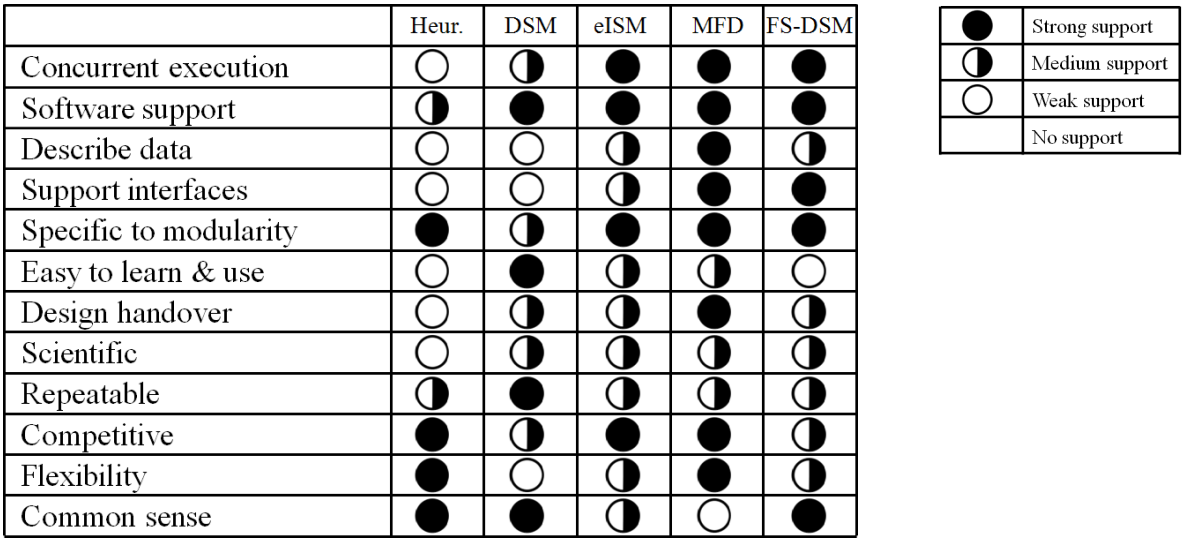

Figure 7. Final scoring of the five methods

Concurrent execution is possible, to some extent, with matrix-based Methods. In MFD, for example, QFD and DPM scoring can be done by parallel teams, once the PPs are determined. Application of Heuristics depends on a function structure diagram. Before such a diagram is created, it is quite hard to apply any of the rules.

Software support is possible with all methods, but as discussed above, Heuristics relies on a function-structure diagram which involves manipulation of graphs, and may be particularly disadvantageous in large projects.

Describe data. MFD describes customer requirements and strategy, as we have seen. FS-DSM does capture strategy, but not requirements. eISM features neither but is unique with its "soft interactive" which increases its score somewhat. Heuristics and DSM are weaker for these three data types.

Support interfaces. MFD and FS-DSM have specific features to support interface generation (make sure modules are clean in terms of strategy and either properties or functions). FS-DSM is stronger in the way it models interactions, but it lacks Product Properties.

Specific to modularity. DSM grew out of a need to plan the sequence of design activities in large projects. The other four methods have specific features to support the generation of modular architecture.

Easy to learn & use. Heuristics relies on a set of rules. One study [Hölttä-Otto 2005] showed it is actually quite difficult to apply the Dominant flow and Branching/Combining rules consistently. In FS-DSM, the algorithm for generating modules is complex and not easy to understand.
Design handover. MFD is the only method that deals consistently with product property goal values, which is an important input in design. MFD and FS-DSM deal with strategic considerations which influences make/buy decisions, among other things. eISM captures "soft interactive requirements" which is shown to be important in design of certain types of products [Sellgren, Andersson 2005]. DSM naturally captures the design sequence.
Scientific. Heuristics is well supported by empirical research on hundreds of real products, but its theoretical foundation is not as clear as the other methods.
Repeatable. Because of its simplicity, DSM is the only method that receives a full score here. The other methods are believed to be roughly equal in terms of repeatability.
Competitive. Full score here indicates the method captures something that is unique to that method. Heuristics does a good job of describing the underlying physics of the product. MFD is unique in its treatment of customer requirements. eISM, as we have seen, is strong on "soft interactive". FS-DSM is based on integration of MFD and DSM and offers nothing truly new (except the integration itself).
Flexibility is lower in DSM because of the fixed TS-TS format. Strategy may be incorporated, as in FS-DSM, but it requires an additional matrix, so it is no longer a pure Component-based DSM.
Common sense. Heuristics, Component-based DSM, and FS-DSM all capture physics (Flow). In addition, FS-DSM captures spatial product considerations.

5. Conclusions

The dendrogram-based approach outlined in this paper provides a systematic way of integrating evaluation criteria from academia with experience-based criteria. Dendrograms allow us to select a suitable number of clusters, depending on the level of detail required.
Derived approaches seem *at least as strong* as the methods, on which they are built. FS-DSM is stronger than MFD in the way it deals with spatial considerations, and stronger than DSM in its integration of strategic considerations. However, module generation is more complex than in either of the original methods. Similarly, eISM successfully captures "soft interactive", absent in both DSM and MFD, but sacrifices both property goal values and strategic considerations (present in MFD). Do derived methods offer improvement, then? Yes, but they suffer from new disadvantages, absent in the original methods.

6. Discussion

Comparisons that include experience-based criteria cannot be completely objective, but the process of *first* determining criteria based on external sources, and *second* scoring the methods on these criteria avoids the problem of a completely opinion-based analysis.
For large projects, where ***Describe data***, ***Software support***, and ***Concurrent execution*** may be important, the matrix-based methods scored higher than Heuristics, which might be better suited to small projects where ***Flexibility*** is valued. Of the matrix-based methods, pure DSM is strong on ***Easy to learn & use*** but lacks features present in MFD, eISM or FS-DSM (see Data types in Figure 1).
In the author's industrial experience, the potential value creation of a new product architecture project is often assessed in a pre-study, where estimated unique part number count reduction is the basis of a financial business-case to determine whether the project is likely to capture sufficient value.
Can we foresee any potential paths to new hybrids? MFD is open enough to accommodate new data types, so extending MFD with spatial or "soft interactive" properties might be one path. Another would be to extend eISM with strategy, essentially by adding a MIM (see Figure 3) to the ISM (see Figure 5). Module generation would probably be more complex.

Acknowledgements

The author wishes to thank the following people for valuable feedback on this paper:
Dr. Ulf Sellgren (Dept. of Machine Design, Royal Institute of Technology, Stockholm), Dr. Gunnar Erixon (Sweden Modular Management AB), and Dr. Katja Hölttä-Otto (Dept. of Mechanical Engineering, University of Massachusetts Dartmouth).

References

Blackenfelt, M. W., "Managing complexity by product modularisation - Balancing the aspects of technology and business during the design process", Doctoral Thesis, Dept. of Machine Design, Royal Institute of Technology, Stockholm, 2001.

Erixon, G., "Modular Function Deployment - A Method for Product Modularisation", Dept. of Manufacturing Systems, Royal Institute of Technology, Stockholm, 1998.

Gilovich, T., Griffin, D., Kahneman, D., "Heuristics and Biases – The Psychology of Intuitive Judgement", Cambridge University Press, Cambridge, UK, 2002.

Huang, G. Q., "Developing Design For X Tools", Design For X - Concurrent Engineering Imperatives, Huang, G. Q. (ed), Springer, New York, 1996.

Hölttä-Otto, K., "Modular Product Platform Design", Doctoral Dissertation, Dept. of Mechanical Engineering, Helsinki University of Technology, Espoo, 2005.

Keller, A., Binz, H., "Requirements on Engineering Design Methodologies", Proceedings of the 2009 International Conference on Engineering Design (ICED09), Stanford, 2009.

Kurfman, M., Stock, M. E., Stone, R. B., Rajan, J., Wood, K. L., "Experimental Studies Assessing the Repeatability of a Functional Modeling Derivation Method", ASME Journal of Mechanical Design, 125(4), 2003, pp. 682-693.

Sellgren, U., Andersson, S., "The Concept of Functional Surfaces as Carriers of Interactive Properties", International Conference on Engineering Design (ICED 05), Melbourne, 2005.

Stone, R. B., Wood, K. L., Crawford, R. H., "A Heuristic Method to Identify Modules from a Functional Description of a Product", Proceedings of 1998 ASME Design Engineering Technical Conferences (DETC98), Atlanta, 1998.

Ulrich, K. T., Eppinger, S. D., "Product Design and Development", McGraw-Hill Eduction Asia, Singapore, 2008.

Zamirowski, E. J., Otto, K. N., "Identifying Product Portfolio Architecture Modularity Using Function and Variety Heuristics", Proceedings of the 1999 ASME Design Engineering Technical Conferences (DETC99/DTM-8760), Las Vegas, 1999.

Corresponding author:
Fredrik Borjesson
Ph.D. student
Department of Machine Design, Royal Institute of Technology (KTH)
Brinellvägen 85, SE-100 44 Stockholm, Sweden
VP of Technology, Modular Management USA, Inc.
8030 Old Cedar Ave S, Suite 203, Bloomington, MN 55425-1215, USA
Telefax: +1 952 854 5586
Email: casachaib@hotmail.com