Publication II

Holtta K. & Salonen M. Comparing three modularity methods. *In Proc of ASME Design Engineering Technical Conferences*. Chicago, IL. September 2-6, 2003.

Proceedings of DETC'03
ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference
Chicago, Illinois USA, September 2-6, 2003

DETC2003/DTM-48649

COMPARING THREE DIFFERENT MODULARITY METHODS

Katja M. M. Holtta
Helsinki University of Technology
Department of Machine Design
P.O. Box 4100
02015 HUT
Finland
Tel +358 9 451 5072, Fax +358 9 451 3549
Email: Katja.Holtta@hut.fi

Mikko P. Salonen
Helsinki University of Technology
Department of Machine Design
P.O. Box 4100
02015 HUT
Finland
Tel +358 9 451 3575, Fax +358 9 451 3549
Email: Mikko.Salonen@hut.fi

ABSTRACT

Modularity has many advantages but few methods exist to partition a product into modules, and no knowledge on which method to use and when. This study compares the heuristic method, the design structure matrix, and modular function deployment. The methods are applied on four commercial products. The disturbing result is that, given the same inputs, all methods partition the products differently. Some consider functionality and interface simplicity whereas others focus on strategic factors. A method developed for single products is poorly suited to modularizing product families. The methods find almost no common module boundaries that would be ideal regardless of the method used. This result is not due to poor application of the methods: the repeatability of each method was analyzed. It varied from 68% to nearly perfect. Modular function deployment was the least repeatable whereas the computerized design structure matrix the most repeatable. The heuristic approach fell in between.

KEYWORDS: modularity, product architecture

INTRODUCTION

Modularity, using structurally independent modules to form a product architecture, is a widely accepted approach to save development and manufacturing costs and bring flexibility to a design. In the recent design literature, there has arisen several methods to modularize product architectures, however, it is not clear which method is most suitable for a specific case. The existing methods have each been optimized for different purposes – some to suit specific customer needs, some to minimize iterations during design stage, etc. [1]. This article looks into three modularity methods, the function structure heuristic method first introduced by Stone *et al*. [2], clustering a design structure matrix (DSM) [4, 5], and Ericsson's modular function deployment (MFD) [3], that have been accepted and used in industry.

The three methods are tested on four products and the results of each method are compared. The methods are applied on four commercial products: an intraoral camera, an electronic pipette and two medical injector heads that belong to the same product family. This is done to test how the methods work with both individual products and product families. A presumption in testing all the methods is that even though all methods arose from different research groups and have different mechanic, they will find some common boundaries – boundaries that are good module interfaces. Also, the repeatability of each method is tested to show possible subjectivity of the methods. The modularity methods, as well as their repeatability, are tested on four products starting from a common functional decomposition. The study will provide insight on the differences between the methods and on which modularity method is the most suitable for a company's specific need.

The remainder of the paper is constructed as follows. The proceeding section will discuss the advantages of modularity. The next section will shortly introduce each of the three modularity methods to be tested. The next section will describe how the methods were used and on what criteria they were compared. Then, the results of applying the methods on first single products and then on a product family are described. The paper will end with conclusions and recommendations.

MODULARITY

The module definition used in this article is adapted from various sources [3, 4, 6]. A module is a structurally independent building block of a larger system with well-defined interfaces. A module has fairly loose connections to the rest of the system allowing an independent development of the module as long as the interconnections at the interfaces are well thought of.

ADVANTAGES OF MODULARITY

The purpose of modularity may be to obtain scale and scope advantages, economies in part sourcing and testing, as well as support mass customization by using common components across a product or a product family [6, 7]. Modularity also makes the design more flexible by allowing independent development of each module and by enabling product variations by changing modules without changes to the rest of the product. [3, 4]

Modularity can also ease the management of product architecture [3]. Blackenfelt [8] describes product complexity by the number and type of elements and relations in a product. In a given architecture the number and type of elements is set, but one can affect the number and type of relations with wise modularity choices. Understanding the interactions between components, modules, and sub-systems also helps the company make sound strategic decisions on e.g. component outsourcing [9, 10]. Modularity decisions can also be made earlier in the technology and research phase. This helps the company decide what physical principles should be investigated [11].

Modularity can be used to simplify the product architecture of a single product and ease the management of design. There is also literature on using modules across products in product families [3, 11, 12]. Modules can be used to provide variety by creating a family of products by using different combinations of modules. Modularity can also ease the design of a product family since the same components are used in more than one product and there is no need to redesign them.

Modularity has also disadvantages. It can mean tradeoffs with performance [7] or excess cost if a low functionality component is replaced by a costly high functionality component in order to make the previously different components the same [13, 14]. Modularity can also have different effects depending on the stakeholder [1] e.g. a supplier's cost might actually rise when the manufacturer applies a certain module regime to reduce its costs.

MODULARITY METHODS

There are only a few well-known systematic modularity methods: so called function structure heuristic method [2, 12], clustering a DSM [5] and MFD [3]. The following sections will introduce the three methods tested in this study.

Function Structure Heuristic method

Function structure heuristic method, developed by Stone *et al*., is based on Pahl and Beitz's function structures [15]. A function structure is a functional decomposition block diagram of all the product's functions and material, energy, and information flows between them. Stone *et al*. separate modules from a single product's function structure by finding the dominant flow, branching flows, or conversion-transmission function pairs [2]. Zamirowski and Otto present three additional heuristics to find common modules across products in a product family. They find similar and repetitive functions within a single product, common functions across products, and unique functions that are found only in one product within the product family and separate them as modules. [12] In addition to these, McAdams *et al.* separate causally linked function pairs as modules [16], but this modularity heuristic is very solution bound and left out of this study. A good tutorial of the method is given by Otto and Wood [17].

The approach is to start with a function structure, and then consider the many possible alternative modules that can be defined by group functions according to the heuristics. The heuristics define possible modules, it is up to the designer to choose which ones make sense. Further and not commonly understood, the heuristics are maximal heuristics. They state only that one should not define modules larger than indicated. Any module defined by a dominant flow as a serial chain of functions, for example, can be subdivided in any way and still be consistent with the heuristics. As such, the approach provides modularity suggestions only, it is not a unique algorithm. Therefore, designer insight and good judgment can enter the process; this is either a benefit or a problem, depending upon one's perspective.

In any case, these heuristics apply to single products and a couple to product families of similar products. The main modularization criteria considered in the function structure heuristic method are functionality and module interfaces. Other criteria such as business or strategy related factors are not represented in the function structure heuristic method, but instead enter through designer judgment in where the rules get applied.

Design Structure Matrix

The DSM [4] can be used to organize product development tasks or teams to minimize unnecessary rework and thus help manage and speed up the development process. The DSM can also be used to define modules within a single product's architecture. In the component or function based DSM, also called architecture DSM, components or functions are placed on the row and column headers of the matrix. Components or functions are then mapped against each other and their interactions are marked in the matrix. One can also present spatial, energy, information, and material interactions of components or functions in a DSM as shown by Pimmler and Eppinger in [5] and also by Blackenfelt in [8]. The interactions can be represented with coupling coefficients -2, -1, 0, 1, or 2 depending on the strength of the relation and whether the relation is beneficial or undesired.

Once functions or components and their interactions are placed in the DSM, a clustering algorithm can be applied to group the functions or components so that the interactions within clusters are maximized and between the clusters minimized. The formed clusters are possible module candidates. There are many algorithms and one can develop one's own to suit the needs of a specific case. The basic idea of a clustering algorithm is to reorder the rows and columns so that all marks are as close to the diagonal as possible or form a tight cluster with other marks. The algorithm used here is developed by Thebeau [18]. This was chosen because it is a well defined computerized algorithm. The algorithm can result in overlapping modules or it may leave a function out of the final clustering, in which case it is up to the designer to decide how to deal with them. The overlapping section could be for example duplicated and placed in both modules or forced to be only in one of the modules where the algorithm suggested it could be. For more about the component based DSM method, refer to [19].

The DSM is designed especially for complex product architectures. The method concentrates on the interfaces of the modules to simplify the design process and the apparent complexity of the product architecture. The component based DSM could be combined with the task and team DSMs to include the modularization in the rest of the design process planning. The method leaves more business oriented factors and product functionality up to the designer's judgment after first simplifying the architecture.

Modular Function Deployment

Another modularization method, perhaps more management- and less engineering-oriented, is MFD [3]. It is also based on functional decomposition, but in this method, modularity drivers other than functionality are considered. MFD is designed to modularize a single product. There are twelve modularity drivers in MFD. Blackenfelt [8] introduces a broader list of 23 drivers, but only the original twelve are discussed here. The first is *carryover* i.e. a specific function will carry over to different products and no technology changes are expected. The next two, *technology evolution* and *planned product changes*, take both unexpected and expected changes into account. *Different specification* enables product variation and *styling* considers how the modularity choice would affect the appearance of the product. *Common unit* is similar to Zamirowski's common function heuristic in the function structure heuristic method. *Process and/or organization*, *separate testing*, *supplier availability*, and *service and maintenance* are related to the organizational effects of modularization. *Upgrading* allows future additions to the product. *Recycling*, the last modularity driver, considers the afterlife of the product. One or a few modularity drivers are chosen according to the firm's strategy. Ericsson and Erixon offer a good tutorial on the method [3].

MFD is similar to Quality Function Deployment (QFD) [20] but here modularity drivers are mapped against functions instead of customer requirements in a matrix. The grouping into modules is started by the functions receiving the highest summed scores; and the functions dominated by the same modularity drivers are good candidates for a module according to this method. Stake [23] and Blackenfelt [8] show how MFD and DSM can be integrated in the grouping phase. Blackenfelt builds a strategic DSM using simplified modularity drivers from the MFD. He suggests using also a functional DSM in conjunction with the strategic MFD [8] to systematize the grouping phase in the MFD. The original method will be used in this study.

MFD suggests that the ideal number of modules is approximately the square root of the number of parts or assembly operations. The estimate is based on optimizing the assembly lead time of the whole product. [3]

In addition, MFD has a step for interface design that considers form, fixation principles, number of contact surfaces and attachments, as well as the number of energy connection points, material flow, and signals. It relies more on engineers than to present a systematic method to locate and choose cut-off points for modules, which again is either a benefit or a problem, depending upon one's perspective.

APPLYING DIFFERENT MODULARITY METHODS

The goal of this study is to analyze modularity methods for single products as well as product families. Four products were chosen for this analysis. The products are an intraoral camera, an electronic pipette, a computed tomography (CT) injector head, and a magnetic resonance imaging (MRI) injector head. The first two are manufactured by Finnish companies. The latter two are produced by an American manufacturer and belong to the same product family. This selection allows comparison of the four modularity methods applied on single products, a product family, and different products at the same time. Also the complexity of the products varies a little allowing the testing of how the complexity of a product architecture affects the usability of the methods. The simplest product, the intraoral camera, has only 15 parts, the pipette 30 and the two injector heads about 70-80 parts.

METHODOLOGY

The study started by decomposing all the product architectures and building function structures. We first built functional block diagrams (function structures) for the product architectures. We decomposed the products to the assembly level of the manufacturer. We assigned each component or sub-system a function e.g. a motor was named "convert electricity into rotation". We then represented all the connections between the components or sub-systems with material, energy, and information flows. For example the motor torque going to the transmission was represented with an (mechanical) energy flow of torque/rotation. As discussed by Otto and Wood [17], one should not forget supporting functions and flows such as vibration and damping or support of weight when completing the function structure. The law of energy conservation should apply in a complete function structure.

Function structures are not required for the MFD and the DSM methods, but they too start with a functional decomposition. Further, the component based DSM [5] can be assigned the same interactions as Pahl and Beitz's function structures except for an additional spatial relationship that is added to the DSM cells. This difference can be removed when one takes the approach of Otto and Wood [17] and augments the function structure with force flows of gravity through the function network, thereby creating an isomorphism between a function structure representation and a DSM representation. The chosen algorithm, however, does not separate between different interactions. Furthermore, since Kurfman *et al.* have shown function structures to be reasonably repeatable [21] the function structure and its functional decomposition, identical for each method, are used as a starting point for all three methods in this study. All products are decomposed to approximately the same level and the same functional basis terms [22] are used when possible. The level of decomposition is as recommended in the MFD [3] i.e. not all the way to nuts and bolts. These actions enable fair comparison of the methods.

The function structure representation of a product is all that is needed to apply the function structure heuristic method. For the DSM method, the function structures need to be converted further into a design structure matrix. This is done by listing the product functions as headers in the rows and columns of a matrix. Interactions between functions are marked in the corresponding intersecting cells of the matrix. For the MFD method, the function structures are converted in a similar manner to a so called module indication matrix. In a module indication matrix, the column headers represent the product functions, whereas the row headers represent the various module drivers for modularizing the product. The next step of MFD, where module drivers are mapped against the product's functions, was done together with engineers familiar with the company and the product in question. This expert filled matrix was used to apply the modularization rules of MFD in the experimental setup explained next.

A total of 20 engineers and graduate students in engineering with either a Bachelor's or Master's degree in Mechanical Engineering at Helsinki University of Technology were used to apply the different modularity methods. They were all well familiarized on each modularity technique by giving them an informative reading package a week before the actual modularization session. The methods were also overviewed right before the experiment and it was ensured that each person felt comfortable with the methods. The actual experiment, modularization of the products, was completed thereafter.

In case of the function structure heuristic method, the engineers and graduate students were given the products' function structures. In addition to their prior familiarization to the method, they were allowed to keep the reading package. Each engineer and graduate student completed the modularization of the products independently, without any external influence on the modularization choices. If questions arose, the subjects were guided only by instructions relating to the methods, not on how to perform the modularization choices. This was done to ensure that the given instructions did not have an effect on the results of this research.

In case of the DSM method, a computer algorithm was used to perform the modularization of the products. This possibility does not exist in the other modularization methods. Due to the possibility to use a computer algorithm on the design structure matrices of the products, and since the results of the computer algorithm are not dependent on the person running it, it was unnecessary to have the subjects manually perform these modularization choices.

In applying the MFD, an identical approach to that of applying the function structure heuristic method was used. The subjects of the experiment were given the expert filled module indication matrices. Each engineer and graduate student thereafter completed the modularization of the products independently, without any external influences on the modularization choices.

In the following sections, each modularization scheme for each product is a compromise among all the engineers, weighing the more experienced engineer's suggestions over the more novice students' results.

The use of several people also allows for calculation of the repeatability of each method. The repeatability was calculated for each method as follows. The modularity schemes of all participants were gathered and the most common solution was formed. Each engineers' or engineering students' answers were compared in deviation to the common solution by calculating the percentage of functions modularized differently. For an overall repeatability metric for the method, the percentages were averaged and subtracted from 100% resulting in a repeatability percentage of each method.

MODULARIZING SINGLE PRODUCTS

The three modularity methods were applied on all four products. Only the dominant flow, branching flows, and conversion-transmission function pair heuristics were applied of the function structure heuristic method, since the other three heuristics are meant for product families only. Note that some of the modules suggested by the heuristics overlap or are alternative options. Tables 1-4 present the results of modularizing the four products.

As assumed, all methods give different suggestions for possible modules. This is due to the fact that each method has been designed to optimize a different factor. The function structure heuristic method minimizes interactions between the modules and finds conversion components such as motors and plungers. The DSM method also minimizes the module interactions but does not separate main flows as the function structure heuristic method does. The MFD, on the other hand, looks into a few chosen strategic factors and leaves module interaction choices to the designer.

In Table 1, all methods recognize the functions *convert electricity to light* and *transmit light* in the intraoral camera as a module of two functions. The function structure heuristic method separates the two functions as conversion-transmission pair. The DSM recognizes the interaction between the two, as well as the lack of interaction to any other functions. The MFD resulted in the same module as well. This was not obvious. *Convert electricity* has a strong module driver profile, different from any other functions. *Transmit light* on the contrary has a weak profile, which means that it could easily be combined with any function. It is a coincidence that all methods grouped the two functions identically. This is due to the fact that the function pair is connected only to one another and is bounded by the system boundary at both ends.

Table 1 The intraoral camera's modules (a module is either a continuous block of uniform color or same colored boxes with identical symbols (e.g. **)

Function	Function structure heuristics	MFD	DSM
Convert ele to light			
Transmit light			
Guide light		*	
Translate focus plane			
Cnv light to image			
Measure image res			
Cnv el&info to transl		*	
	#modules: 3+	4	3

An example of a dissimilar module suggestion in the intraoral camera is the different groupings of functions *guide light*, *translate focus plane*, and *convert light to image*. All three functions are identified as a dominant flow by the function structure heuristic method. The MFD method, on the other hand, groups only the first two together leaving the *convert light to image* as a separate module on its own. This is due to the different module driver profiles: *convert light to image* has a high score for example on supplier availability, which the other functions do not have. This difference suggests that the functions are to be grouped in different modules. In addition MFD suggest including *convert electricity and information to translation* with the other two functions. The DSM method does not give a definite answer for the *convert light to image* function. The method suggests it could be either with *guide light* and *translate focus plane* as suggested by the function structure heuristic method, or with *measure image resolution* function.

Table 2 shows similar results for the electronic pipette. For example, all methods identify parts of the drive unit (*convert electricity to rotation…convert rotation to translation*) as a module candidate, but the methods do not agree on which functions should exactly be included in the module.

Table 2 The electronic pipette's modules (a module is either a continuous block of uniform color or same colored boxes with identical symbols (e.g. **).

Function	Function structure heuristics	MFD	DSM
Control pipetting			**
Convert e to rot		*	***
Change rot			
Transmit rot			
Indicate position			
Convert rot to trans			
Reg cylinder vol			
Store sample			
Convert e to magn			**
Stop rotation			
Posit rot preventer			
Store torque		*	***
	#modules: 3+	6	5

Yet another example of the difference between the three modularization methods is the drive mechanism, or the functions *convert electricity to torque*, *reduce speed*, and *transmit torque* in the MRI injector (Table 3). These functions are identified as a part of a dominant flow by the function structure heuristics. The same method also finds *convert electricity to torque* as a conversion type module. MFD on the other hand leaves the *transmit torque* function out of the module. This is because this function has very different values in two major modularity drivers than the other functions. DSM suggests yet another option and combines the *convert electricity to torque* function with the *encoder*. This choice is reasonable since the *convert electricity to torque* (motor) and *encoder* are closely connected. Because the *encoder* is not part of the dominant flow in the function structure, it is thus not identified by the function structure heuristics. Similar results are found in the CT injector example (Table 4).

Table 3 MRI injector's modules (a module is either a continuous block of uniform color or same colored boxes with identical symbols (e.g. **).

Function	Function structure heuristics	MFD	DSM
Supply power			*
Channel power			
Control HCU Connect OTR to HSU			
Cnct HCU,HSU,OTR Connect HSU to HCU			**
Import user input HSU			
Indicate arm status		*	
Change color			
Encoder		•	
Convert e to torque Reduce speed		**	
Transmit torque		***	
Convert torque to transl		**	***
Change syringe volume		••	
Store contrast		•••	
Sense position		•	***
Import user input knob		**	
Indicate arm status		*	
Change color			
Encoder		•	*
Convert e to torque		**	
Reduce speed			
Transmit torque		***	
Convert torque to transl		**	••
Change syringe volume		••	
Store flush		•••	
Sense position		•	**
Import user input knob		**	••
	modules: 4+	8	12

MFD is the only method that limits the number of modules. It is thus more likely to suggest fewer modules than for example the DSM. The function structure heuristic method is not meant to modularize a whole system. It gives suggestions for good module boundaries. The designer must then decide how to group the functions, particularly "chains" of a single function that can be grouped in any way, as according to the dominant flow heuristic. The intraoral camera and the electronic pipette are such simple systems that there is no significant difference between the total number of modules in the architectures suggested by different methods (Tables 1 and 2). More typical differences in the number of modules can be seen in the MRI injector example (Table 3) and in the CT injector example (Table 4).

Table 4. CT injector's modules (a module is either a continuous block of uniform color or same colored boxes with identical symbols (e.g. **).

Function	Function structure heuristics	MFD	DSM
Indicate data			•
Import user input			
Import on/off signal			*
Change acoustic vibr			•
Control injector head			
Connect to HCU			*
Connect syringe			***
Sense 125ml syringe			
Sense 200ml syringe			•
Sense syringe in place			
Increase temperature		*	
Sense position			**
Sense position			
Encoder			
Encoder			
Convert e to torque			
Transmit torque			**
Reduce speed			
Convert torque to transl			
Change syringe volume			
Store contrast			***
Secure motor			**
Secure drive assy			
Indicate arm status			•
Indicate arm status		*	
Convert h.force to rot			
	modules: 7+	7	10

One could presume that all three modularity methods find some common interfaces, interfaces that are ideal at module boundaries since the objective of modularity is to find modules that have simple interfaces to the rest of the system. This however was not the case. The three methods tested identify a few common interfaces between suggested modules. A closer look at these shows, however, that truly common interfaces are only at the product boundaries e.g. *change color* (colored light indicating status to the user) and *store sample*, *contrast*, or *flush* (disposable syringe). Also in the seemingly identical modules identified by all methods in the intraoral camera (*convert electricity to light* and *transmit light*), the module boundaries are actually boundaries to the outside of the system (from a power supply and into a patient's mouth). All three methods grouped the same interfaces to the outside world in the same way.

The only exceptions in all four products, where all methods found the same interface, occurred for no obvious reasons. This occurred at the interface between a connector card and the arm light drives in the MRI injector example and the interface between the motor and the control circuit board in the CT injector. Analysis of these interfaces brings no obvious answers as to why these two interfaces were identified by all methods since there are very similar, practically identical, interfaces that were not identified the same by all the methods: For example, the CT injector head has very similar arm lights to the MRI injector, the only difference is that it is connected to a more complicated control card. Similarly, the MRI injector has also an interface between a control circuit board and a motor, but it is not identified by all methods as in the CT injector.

One clear difference did become clearly obvious to all, and that was the extent of designer judgment allowed to enter the process. The DSM is clearly the most repeatable and the MFD the least repeatable method, when starting from the same functional decomposition as in this study.

After the functional decomposition the DSM gives the same suggestion of possible modules regardless of who runs the algorithm. We chose a well defined computerized algorithm for this study. Having chosen a more heuristic clustering algorithm would have naturally reduced the repeatability considerably. Furthermore, the algorithm can produce overlapping modules (as it did in case of the intraoral camera and the CT injector), which leads to a more subjective choice of how to handle the overlapping section. The final grouping of modules in the MFD, on the other hand, gave different results depending on the designer's opinion. Most engineers found the same cores for the modules but the combining of the rest of the functions when the number of modules was very limited resulted in different results depending on who did it. The heuristics are somewhat more repeatable than the MFD. The application of the conversion-transmission pairs are particularly independent of the designer, due to the clear definition of the heuristic. Different designers also tended to agree upon branching flows. The dominant flow, however, is the most vulnerable to subjective choices. Similar results were also shown by Kurfman *et al*. in their experiments [21]. Table 5 shows the repeatability of each method applied on a single product.

Table 5 Repeatability of modularity methods.

Method		Repeatability
Function structure heuristics	Dominant Flow	75%
	Branching Flow	80%
	Conversion-Transmission	90%
DSM		100% *
MFD		68%

*Algorithm run by a computer

FINDING COMMON MODULES ACROSS PRODUCTS

Common modules across different products are needed for platforming product families. To find common modules across the two injectors that belong to the same product family, product family heuristics were applied in addition to the modularity methods already applied to modularize a single product. The DSM and the MFD identified almost no common modules across the products. This is because both methods have been optimized for modularizing a single product. The DSM found similar clusters of the motors and encoders and the MFD recognized the disposable syringe (*store contrast/flush*) as a common module from both injectors.

The function structure heuristic method identified almost equivalent clusters for the drive mechanisms (dominant flow) and it also suggested that the arm lights are similar in both injectors (branching flow). When we added the product family heuristics, we found a few common functions that could be shared across the two products (Table 6). A common function chain that could possibly be shared between the two injectors is *convert electricity to torque*, *reduce speed*, *transmit torque*, *convert torque to translation*, *change syringe volume*, and *store sample/contrast/flush*. In addition, also other commonalities were found by the heuristics. Functions *indicate arm status*, *sense position*, and the *encoders* were recognized as common module candidates between the two injectors.

Table 6 Modules identified with the product family heuristics in the MRI and the CT injector. A module is formed by boxes with uniform color (within the product by means of "similar" heuristic) or by boxes with identical symbols (between the products by means of "common" heuristic). Unmarked functions form unique modules.

MRI injector's modules identified by product family heuristics		CT injectors modules identified by product family heuristics	
Supply power		Indicate data	
Channel power		Import user input	
Control HCU		Import on/off signal	
Connect OTR to HSU		Change acoustic vibr	
Cnct HCU,HSU,OTR		Control injector head	
Connect HSU to HCU		Connect to HCU	
Import user input HSU		Connect syringe	
Indicate arm status	**	Sense 125ml syringe	
Change color		Sense 200ml syringe	
Encoder	x	Sense syringe in place	
Convert e to torque	*	Increase temperature	
Reduce speed	*	Sense position	¤
Transmit torque	*	Sense position	¤
Convert torque to tra	*	Encoder	x
Change syringe vol	*	Encoder	x
Store contrast	*	Convert e to torque	*
Sense position	¤	Transmit torque	*
Import user input knob		Reduce speed	*
Indicate arm status	**	Convert torque to tra	*
Change color		Change syringe vol	*
Encoder	x	Store contrast	*
Convert e to torque	*	Secure motor	
Reduce speed	*	Secure drive assy	
Transmit torque	*	Indicate arm status	**
Convert torque to tra	*	Indicate arm status	**
Change syringe vol	*	Convert h.force to rot	
Store flush	*		
Sense position	¤		
Import user input knob			

The repeatability of the product family heuristics is of the same order then the repeatability of the other heuristics (see Table 7) The fact that the common unit heuristic is the least repeatable is explained by the fact that similar and repetitive as well as unique functions are searched within a product and common units across products. There is more room for error and interpretation when comparing not only functions within a product but across products. Using the standard wording of functional basis improves the repeatability, but many functions were still a matter of engineer's opinion.

Table 7 Repeatability of product family heuristics.

Method		Repeatability
Function structure heuristics	Similar/Repetitive	81%
	Common	70%
	Unique	86%

The results presented here cannot be claimed universally applicable, since the scope of the study remained small and only in one industry. It would be practically impossible to be able to have a statistically significant amount of products for this modularity method analysis since they, especially the MFD, require detailed information about their manufacturing companies' strategies etc. This study, however, still provides good insight into how different modularity methods differ from one another, when to use them and especially how repeatable each method is. And we believe that the results can be extended to other products as well as other industries.

CONCLUSIONS

In this study three modularity methods, as well as their repeatability, were tested on four products starting from a common functional decomposition. It was shown that all three methods, the function structure heuristic method, the DSM and the MFD, given identical inputs, produce different results. This is due to the different viewpoints and application areas of each method. The function structure heuristic method considers the functionality of the product and interface simplicity whereas the DSM considers only the interface simplicity but it can be combined with other strategic matrices to take also other company issues into account. The MFD, on the other hand, focuses on various strategic issues leaving the decisions about the functions and interfaces of the product to the designer.

All three methods were tested on four commercial products: two single products and on a product family of two medical injector heads. It is notable that the methods designed for single products are poorly suited to modularizing a product family for platforming.

It was presumed that even though the methods are very different, they should find some common module boundaries in a product since the fundamental goal of each method is the same. This however was not true and leads to a conclusion that no method is perfect and the choice of method to use depends on the case in hand. The MFD is best suited for strategy based modularization, to define design variants and decide on buy-make decisions, for example. To decide on the exact module boundaries i.e. to minimize the interactions at each boundary the function structure heuristic method is the best choice. It helps the engineers design fairly independent modules, that is, modules that if changed, they will not affect the rest of the system too much. The DSM can also be used to simplify the module boundary interactions. It is best suited to modularize a more complex system where there are too many interactions for a person to handle. The DSM can also be used for organizing product development teams and tasks. Another difference between the DSM and the function structure heuristic method is that the former, being a computerized algorithm, can handle complex problems quickly but may suffer from lack of the flexibility and reasoning of a human mind and suggest some irrational modules. To modularize a product family, the family heuristics are the only reasonable choice, since they are the only tools designed for that purpose. The MFD has a common unit driver to handle product family issues, but it is only one of many drivers.

The cases when one should apply each method are different, which leads to a conclusion that none of the methods should probably be used on its own. A solution could be to start with one method and check additional factors with another method. One could for example run an MFD on a product and use the function structure heuristics to define the final module boundaries. Another example could be to use the product family heuristics on a platforming case to identify common modules across the products and then continue modularizing the rest of each product with another method.

The repeatability of the methods varied from 68% to practically 100%. The MFD was clearly the weakest in repeatability. The DSM on the other hand can be run by a computer and is therefore repeatable. But the algorithms may ask for simplifications in the architecture that are not possible in the real world. The function structure heuristics fall in between in their repeatability. This also means that the MFD is the least favorable method for a complex product and the DSM the most favorable. Again the function structure heuristics are in between the two. Table 8 summarizes the differences between the three modularity methods (note: in this table also the repeatability of the product family heuristics is included in the value related to the repeatability of the function structure heuristics). No method dominates the others. The choice of which method to use and when should be done case by case. MFD is suitable for forming customer modules, the other methods for more technical modules; maybe for sub-modules of customer modules. Choosing a modularity method depends on the modularizing objectives.

Table 8 Summary of the three modularity methods.

	Function structure heuristics	MFD	DSM
Single product	yes	yes	yes
Product family	yes		
Repeatability	70-90%	68%	100%
Considers functionality	yes		
Interface design	yes		yes
Strategic issues		yes	
Organization		yes	yes

ACKNOWLEDGMENTS

The authors would like to thank Helsinki University of Technology Department of Machine Design and MIT Center for Innovation in Product Development for funding this research. We would also like to give our appreciation to Kevin Otto for his valuable insights, and Product Genesis Inc. for helping in our research. Finally a sincere thank you to all the companies that participated in our study.

REFERENCES

1. Fixson, S., 2001, “Methodology Development: Analyzing Product Architecture Implications on Supply Chain Cost Dynamics”, Paper presented at the 5th Conference on Technology, Policy, and Innovation “Critical Infrastructures”, Delft, The Netherlands.
2. Stone, R. B., Wood, K. L. and Crawford, R. H., 2000, “A heuristic method for identifying modules for product architectures”, *Design Studies*, **21**, 1, pp. 5-31.
3. Ericsson, A. and Erixon, G., 1999, *Controlling design variants: Modular product platforms*. ASME Press. New York, NY.
4. Ulrich, K. T. and Eppinger, S. D., 2000, *Product Design and Development*, McGraw-Hill, New York, NY, 2nd ed.
5. Pimmler, T. U. and Eppinger, S. D., 1994, “Integration Analysis of Product Decompositions”, *ASME Design Theory and Methodology Conference*, Minneapolis, MN.
6. Baldwin, C. Y. and Clark, K. B., 2000, *Design Rules: The Power of Modularity* Design, MIT Press, Cambridge, MA.
7. Whitney, D. E., *Mechanical Assemblies: Their Design, Manufacture, and Role in Product Development*, to be published by Oxford University Press.
8. Blackenfelt, M., 2001, *Managing complexity by product modularization*, Doctoral Thesis, Dept. of Machine Design, Royal Institute of Technology, Stockholm.
9. Camuffo, A., 2001, “Rolling Out a “World Car”: Globalization, Outsourcing and Modularity in the Auto Industry”, Working Paper. International Motor Vehicle Program, Massachusetts Institute of Technology.
10. Mikkola, J., 2000, “Modularity, outsourcing, and inter-firm learning”, DRUID Summer conference, Rebild. Denmark.
11. Dahmus J. B., Gonzales-Zugasti J. P., and Otto K. N., 2000, “Modular Product Architecture”, *Proc. of DETC 00: ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Baltimore, MD.
12. Zamirowski, E. J. & Otto K. N., 1999, “Identifying Product Family Architecture Modularity Using Function and Variety Heuristics”, *11th International Conference on Design Theory and Methodology*, ASME, Las Vegas, NV.
13. Gupta, S. and Krishnan, V., 1999, “Integrated Component and Supplier Selection for a Product Family”, *Production and Operations Management*, **8** (2), p.p. 163-181.
14. Krishnan, V. and Gupta, S., 2001, “Appropriateness and Impact of Platform-Based Product Development”, *Management Science*, **47**, 1, pp. 52-68.
15. Pahl G. and Beitz W., 1999, *Engineering Design*. Springer-Verlag, London Ltd, 2nd ed.
16. McAdams D. A., Stone, R. B., and Wood, K. L., 1999, “Functional interdependence and product similarity based on customer needs”, *Research in Engineering Design*, **11**, 1, pp 1-19.
17. Otto, K. and Wood, K., 2001, *Product design: techniques in reverse engineering and new product development*, Prentice Hall, Upper Saddle River, NJ.
18. Thebeau, R. E., 2001, *Knowledge Management of System Interfaces and Interactions for Product Development*

Processes, MIT Masters Thesis, System Design & Management Program.

19. Browning, T. R., 2001, "Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions," IEEE Transactions on Engineering Management, **48**, 3, pp. 292-306.
20. Hauser, J.R., and Clausing, D., 1988, "The House of Quality," Harvard Business Review, p.p. 63-73.
21. Kurfman, M., Stone, R., Van Wie, M., Wood, K., Otto, K., 2000, "Theoretical Underpinnings of Functional Modeling: Preliminary Experimental Studies", Proc. of DETC 00, Baltimore, MD.
22. Hirtz, J., Stone, R. B., and McAdams, D. A., 2002, "A functional basis for engineering design: Reconciling and evolving previous efforts", *Research in Engineering Design*, **12**, pp. 65-82.
23. Stake, Roger B., 2000, *On conceptual development of modular products*, Doctoral Thesis, Division of Assembly Systems, Dept. of Production Engineering, Royal Institute of Technology, Stockholm.

