INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED'09
24 - 27 AUGUST 2009, STANFORD UNIVERSITY, STANFORD, CA, USA

IMPROVED OUTPUT IN MODULAR FUNCTION DEPLOYMENT USING HEURISTICS

Fredrik Borjesson
Royal Institute of Technology (KTH), Department of Machine Design, Stockholm, Sweden

ABSTRACT
In Modular Function Deployment, technical solutions are grouped into modules according to the product properties and the strategic intentions of the company. Statistical methods such as hierarchical clustering are useful in the formation of potential modules, but a significant amount of manual adjustment and application of engineering common sense is generally necessary. We propose a method for promoting better output from the clustering algorithm used in the conceptual module generation phase by adding Convergence Properties, a collective reference to data identified as option properties, geometrical information, flow heuristics, and module driver compatibility. The method was tested in a case study based on a cordless handheld vacuum cleaner.

Keywords: Conceptual product development, modular products, Modular Function Deployment, module drivers, clustering algorithm, hierarchical clustering, statistical approach, heuristic methods

0 ABBREVIATIONS
Table 1 is a summary of the abbreviations used, with a brief description.

Table 1. Abbreviations, existing and proposed

Existing accepted terminology

Data

Abbre-viation	Meaning
CR	Customer Requirement. Expression of customer need.
PP	Product Property. Measurable and controllable translation of CR.
TS	Technical Solution. Physical carrier of a required function.
MD	Module Driver. Describes the company-specific strategy.

Matrices

Abbre-viation	Meaning
QFD	Quality Function Deployment. Defines the relation between CR and PP.
DPM	Design Property Matrix. Defines the relation between PP and TS.
MIM	Module Indication Matrix. Defines the relation between TS and MD.
PMM	Product Management Map. Interlinked visualization of QFD, DPM, and MIM.

Proposed new terminology

Data

Abbre-viation	Meaning
OP	Option Property. Presence of specific technical features.
GP	Geometrical Property. Representation of key regions in product.
HP	Heuristic Property. Application of branching-combining, conversion-transmission, and dominant flow heuristics.
DC	Driver Compatibility. MD compatibility to guarantee strategically compatible clusters.
CP	Convergence Properties. Collective reference to OP, GP, HP, and DC.

Matrices

Abbre-viation	Meaning
QED	Extended QFD. Defines the relation between CR and OP.
CPM	Convergence Property Matrix. Defines relation between CP and TS.
ePMM	Extended PMM. Standard PMM plus QED and CPM.

1 INTRODUCTION

Modularity methods are concerned with the translation of customer requirements and company strategy into a product architecture that offers reduced time-to-market (by allowing flexible configurations), lower unique part number count and often material cost reduction (through reduction of suppliers and improved purchasing leverage). According to Hölttä-Otto [1], there are three main approaches to modularity.

Heuristics refer to rules of thumb that very often give good results. In [1], two main categories are investigated: modules dictated by the patterns of flow (matter, energy, and information) between the functional blocks, and patterns of commonality/variety in a family of products. These methods are highly repeatable [1] but do not consider strategic objectives [2].

DSM [3] may be used to determine the ideal sequence of development tasks in a project, but it can also be used to define modules in a product architecture [1]. The best sequence is one that minimizes the number of coupled tasks. DSM does not consider strategic objectives or even functional requirements of the product.

MFD [4] is a five-step method for translating customer requirements into a modular architecture, while considering the strategic objectives (described using twelve predefined Module Drivers). Cross-functional teams are used. Project data is captured in three core matrices. MFD allows for a high level of concurrency in the conceptual phase, before modules are defined. In this paper, a module is defined as a functional block with standardized interfaces, selected for company-specific reasons [4].

Module generation is based on grouping Technical Solutions into modules with related functions and similar strategic intent. In real projects, this involves sorting a large amount of data, and for practical reasons this must be done using statistical methods. The output is often shown as a dendrogram, a hierarchical representation of the level of similarity between the Technical Solutions. Very often, however, the first dendrogram just does not seem fully to make sense.

This paper is part of a larger research topic, with the goal of determining how we can improve MFD to yield better output. A better understanding of the product properties that drive product architecture decisions is believed to be at the heart of this question. This paper deals with a more narrowly defined topic: the introduction of so called *Convergence Properties* to yield more useful conceptual module output from the statistical algorithms.

2 BRIEF MFD THEORY

Without a solid understanding of Customer Requirements, any product architecture effort risks becoming an engineering-driven exercise without useful market application. QFD [5] is a powerful tool for describing Customer Requirements in terms of Product Properties.

Shortly after MFD [4] was introduced, it was improved by the addition of the Design Property Matrix (DPM) [6], linking the Product Properties in the QFD with the Technical Solutions of the Module Indication Matrix (MIM). This version of MFD is the reference for comparisons with proposed improvements.

Module Drivers are used to describe the strategic intent of an architecture, and is a key feature of the MFD approach. The Drivers are summarized in Table 2 below.

Table 2. Module Drivers used in MFD [4] (continues on next page)

Driver	Technical consideration
Common Unit	Allow solutions to be used in several variants
Carry Over	Allow solutions to be used in future product generations
Technical Specification	Create a range of modules with regard to specification level
Styling	Create a range of modules with regard to styling variation

Table 2. Module Drivers used in MFD [4] (continued from previous page)

Driver	Technical consideration
Planned Design Changes	Allow new design to be incorporated
Technology Push	Allow new technology to be incorporated
Process / Organization	Protect scarce resources in development or production
Strategic Supplier Available	Outsource development and manufacturing to external partner
Separate Testability	Allow module level testing before final assembly
Service / Maintenance	Allow easy replacement or service of parts
Upgrading	Allow customer to upgrade performance after purchase
Recycling	Extract dangerous or valuable materials at time of scrapping

Module Drivers support basic company strategy. Drivers that support the same strategy are said to be compatible [4]. A module should consist of technical solutions with compatible Module Drivers only. Empirical research [7] indicates the interpretation of driver compatibility varies somewhat from one company to another. However, Figure 1 provides a useful guideline. Conflicting driver combinations are indicated by a minus sign while mutually supporting drivers get a plus sign. A question mark means potentially there is interaction, but the nature must be determined on a case-by-case basis.

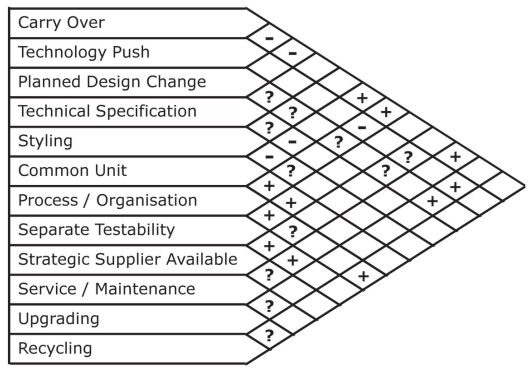

Figure 1. Module Driver compatibility according to Erixon [4]

Some of the strengths of MFD are:

- Allows high degree of parallelism in the conceptual phase of the work
- Use of matrices for all project data implies *automated statistical approaches to module generation* may be used, which is particularly useful in large projects
- Incorporates customer driven, engineering driven, and strategic considerations

Some of the weaknesses of MFD are:

- Results depend on the experience and technical expertise of the team conducting the work [1]
- Quality is highly dependent on good property definitions and consistent scoring, which typically requires experience
- For certain product types, there is not always a sufficient number of customer-driven product properties to generate modules on the right level of resolution

3 CONVERGENCE PROPERTIES

The reason why hierarchical clustering often does not converge on a useful first output is problems with the data, not the algorithm. When engineers apply their common sense to determine what constitutes a useful output, they rely on additional information which typically is not part of the data supplied to the algorithm.

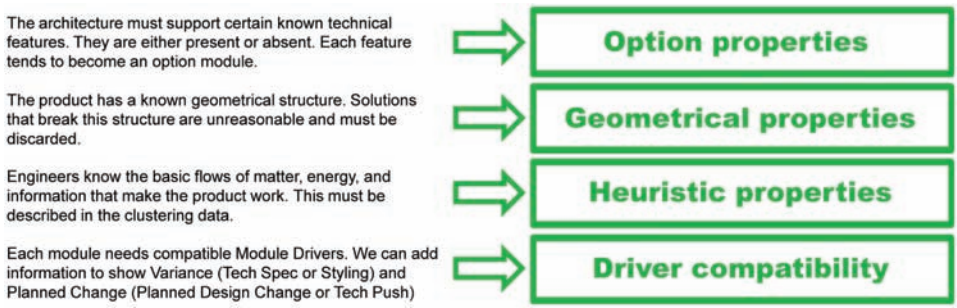

Figure 2. Convergence Properties are a response to common practical issues in MFD

We aim to promote better output in the module-clustering phase of MFD by considering several practical concerns and including data pertaining to the four areas named in Figure 2. The concept of *Convergence Properties* is introduced to formalize some of the common sense applied in practical applications of MFD. In this paper, we will look at four types of Convergence Properties.

Option properties determine whether a certain product option is enabled in the final product configuration. These are special Boolean properties, e.g., take yes/no values only. Traditional Product Properties used in MFD are solution-free. Option properties are radically different in that they normally stipulate a specific solution.

Geometrical properties reflect additional knowledge about the probable physical configuration of the final product. We may imagine dividing the product into pre-defined regions, and then labeling each technical solution by whether it would likely be located in a certain region.

Heuristic Properties are based on function structure. Two sets of heuristics are presented in [8] and [9]. The former are based on flow of matter, energy, and information, whereas the latter are based on function and variety (Causally-linked functions, Similarity/repetition, and Commonality/variety). We will be using the flow heuristics in this work, but not the function and variety heuristics, since the latter overlap more strongly with what may be achieved using the normal Product Properties and Module Drivers of traditional MFD. Flow heuristics describe how functions interact in the product, which complements the performance-based properties of MFD. Each of the three flow heuristics are presented below. The word Flow refers to a flow of matter, energy, or information.

Dominant flow : If the same Flow goes through a sequence of functions, they should form a module.

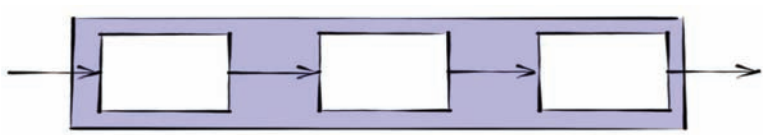

Figure 3. Dominant flow heuristic

In products where there is a strong dominant Flow, this Flow typically determines the system performance. Almost always, the technical solutions have to be in a certain sequence. If the

performance level has to be changed, all the technical solutions involved in that flow typically have to be re-engineered in harmony. A jet engine might be an intuitive example: stepping up the thrust involves changing inlet, fan blades, compressor, exhaust geometry etc.

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.

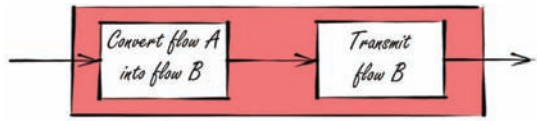

Figure 4. Conversion-transmission heuristic

This heuristic is based on the idea that Flow transformation typically happens in a technical solution that has some critical property related to the conversion itself, and this property tends to be very local to that technical solution. Transmission is included because it is convenient to deliver the output to where it is needed. A good example would be a DC-motor with outgoing shaft.

Branching-combining : If a Flow splits up into parallel function chains, the subfunctions that make up those chains should form modules. This also applies to combining flows.

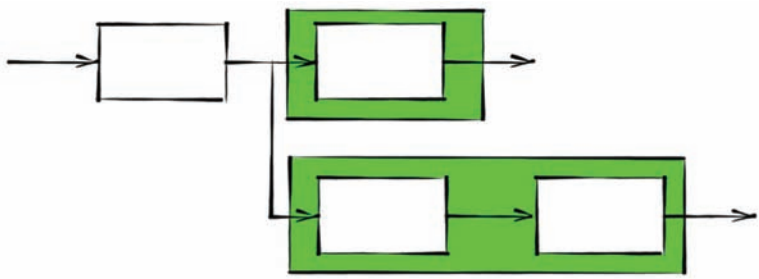

Figure 5. Branching-combining heuristic

Branches dealing with the same Flow often perform independent tasks and often have to be flexibly configurable. To enable that, it is useful to use the same interface in each branch. This results in a bus architecture, as in a PC where boards can be added flexibly.

Driver Compatibility may be used to define groups of Module Drivers in such a way that drivers within a group support the same strategy. Very often in MFD project work, the DPM is much larger than the MIM. Since the clustering algorithm is essentially statistical, the strategic intent reflected in the MIM is drowned out by the performance requirements described in the DPM. By supplying additional information about compatibility, we can decrease the risk that Technical Solutions with incompatible drivers get allocated to the same module.

Extended PMM accommodates Convergence Properties

One of the key outputs of MFD is the Product Management Map (PMM), which in turn consists of three matrices, Quality Function Deployment, Design Property Matrix, and Module Indication Matrix. A traditional PMM is shown on the left in Figure 6. An Extended PMM (ePMM) is shown on the right, and it represents a potential solution to the problem of *incorporating Convergence Properties into MFD in a matrix format to allow usage of statistical tools for module generation*. The ePMM has two additional matrices: the Convergence Property Matrix (CPM) and the Extended QFD (abbreviated QED to underscore the analogy to QFD). The QED uses Option Properties that take yes/no values only. The remaining Convergence Properties are not derived from customer needs, so that part of QED is left blank.

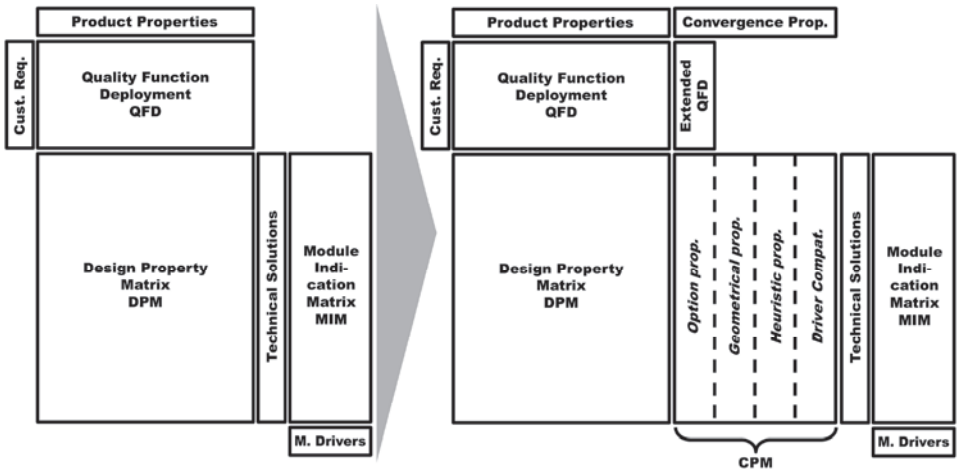

Figure 6. PMM and the Extended PMM (ePMM)

4 CASE STUDY – CORDLESS HANDHELD VACUUM CLEANER

Background

The case study presented here is based on a cordless handheld vacuum cleaner, which has been used successfully in a basic five day MFD-training for hundreds of students. This case study is based on learnings from that training as well as desk research. One key element of the training is the scoring of the DPM and the subsequent application of clustering on that data, which is shown schematically in Figure 7.

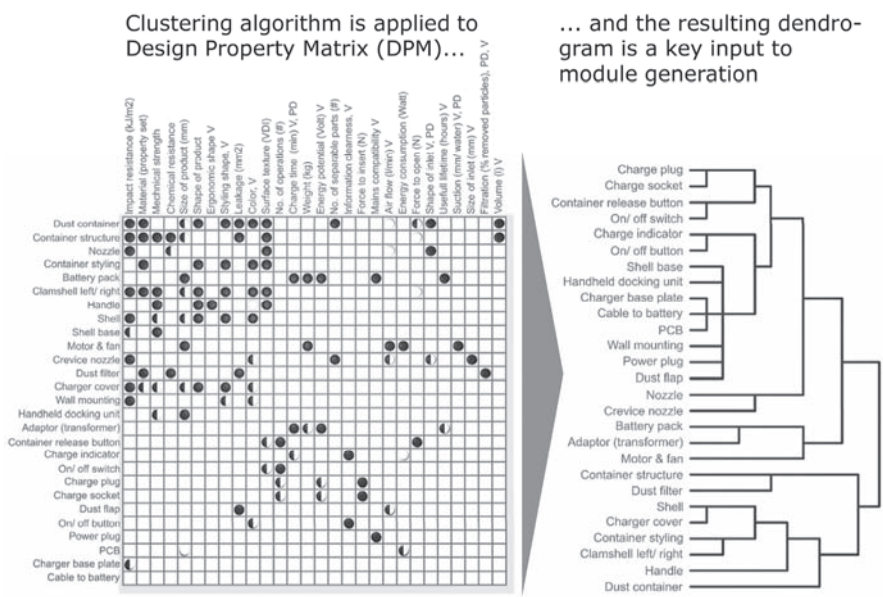

Figure 7. Schematic of the transformation of DPM into a dendrogram

Figure 8 shows how a team might interpret the dendrogram in Figure 7. Potential modules are indicated with boxes.

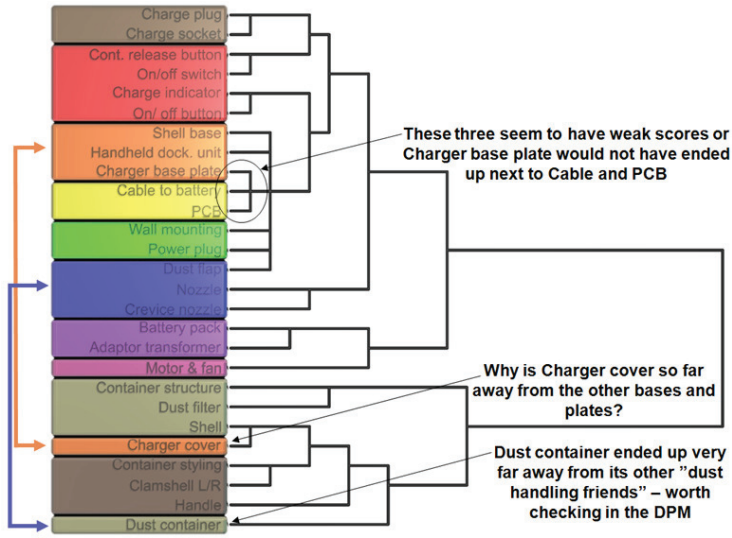

Figure 8. Possible interpretation of the dendrogram in Figure 7

Project experience has shown that dendrograms rarely can be used as is. In the example above, there are several irregularities. This raises the suspicion that our example DPM contains incorrect scoring or even poorly defined properties.

Why dendrograms must be interpreted

Table 3 shows why dendrograms often have to be interpreted, and not used as is.

Table 3. Examples of reasons why dendrograms need to be interpreted

Reason	Example
Geometrical information about the product is not considered	All product styling elements, regardless of physical location, become one single module
Critical properties are overlooked	Two different fuel sources, electricity and gas, are treated the same, resulting in impossible dual-fuel modules
Engineering properties cloud the DPM matrix	Too much weight is placed on surface and material properties, resulting in a skewed QFD and not enough focus on interface-driving properties
Required sequence of certain technical solutions is ignored	Nozzle and exhaust create a module, ignoring the need for an impeller to drive the flow
Modules are not strategy clean	Size of DPM overshadows MIM completely, resulting in a clustering that is almost completely DPM-based
Important requirements are not translated into useful properties	"Easy to clean" is translated into a statement such as "cleanability", which on the surface may look like a property, but in reality is nothing but a reworded requirement
Key concept selections are treated as any other property	The method used to release a dust container from the handheld unit is captured by a property-like statements such as "Number of steps to remove container"
Tremendous energy is expended on engineering properties	Color, finish, and material are described in tremendous detail, when normally surface properties do not drive an interface in the architecture
System properties are never disaggregated	"Suction power" is introduced, resulting in modules that are simply much too large, vaguely corresponding to subsystems

To show how the proposed four types of Convergence Properties might help module creation in the handheld vacuum cleaner example, we will now look at an example of each one.

Option Properties

When we devise a new architecture for a family of cordless vacuum cleaners, we may decide to support an optional charge in progress indicator, as shown in Figure 9. We may capture this by creating an Option Property called "Presence of Charge Feedback".

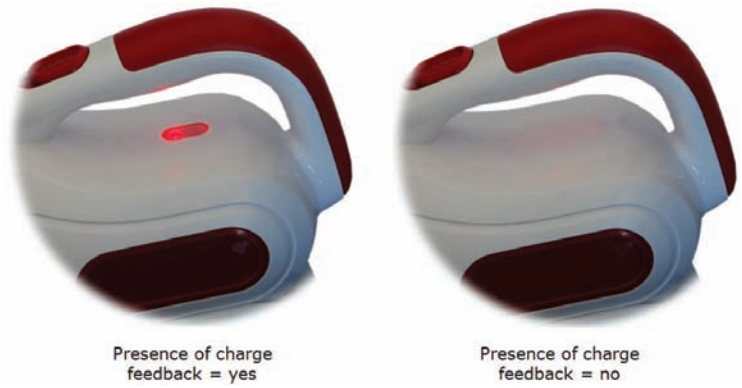

Figure 9. Charge feedback is either present or it is not. Photo manipulated by author.

Geometrical Properties

Without knowledge of product geometry, hierarchical clustering could propose a module composed of suction nozzle and air exhaust, for example. They both relate to pressure drop and are both characterized by an important cross section. Such a module would ignore the normal physical configuration of the product. In terms of geometry, we know the nozzle is typically in the front, and the exhaust typically in the back. Figure 10 shows three regions: Front, Rear, and Off unit.

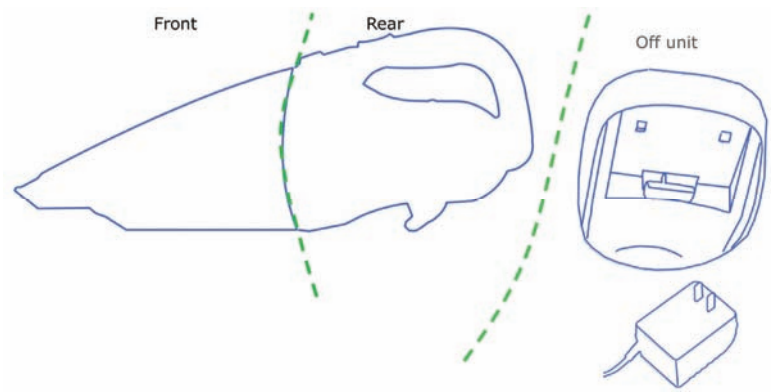

Figure 10. Geometrical properties describe the region where a TS belongs

Heuristic Properties

The dominant flow heuristic is particularly suitable in a handheld vacuum cleaner, where there are flows of air and energy. This heuristic predicts that any technical solution involved in the main airflow should be part of one and the same module. We create a Heuristic Property called Dominant Air Flow, and tag all the involved technical solutions in CPM. This holds them together in the clustering.

Driver Compatibility

In this example, two groups were introduced, Variance and Planned Change, defined as follows

- Variance: enforce distinction between Common Unit and either Styling or Different Specification
- Planned Change: enforce distinction between Carry-over and either Planned Design Change or Technical Evolution

If a Technical Solution had a MIM score on Styling or Different Specification, it would also receive a score on Variance. Similarly, Planned Design Change or Technical Evolution would give it a score on Planned Change.

Putting CPM to the test

We compared a traditional PMM with an ePMM, and examined the dendrogram output. The Product Properties and Convergence Properties that were used are summarized in Table 4. The Module Drivers were used as well and they can be found in Table 2.

Table 4. Variables used in analysis

In MFD	Type	Name	Meaning
DPM	Product Property	Scratch resistance	High for outdoor applications
		Container volume	Dust storage volume
		Battery Capacity (mAh)	Battery capacity
		Particle size	Largest particle through filter
		Nozzle Angle	Pointiness of nozzle
		Output filter (mm^2)	Exhaust diffuser area
		Resolution of speed control	Type of movement of slider
		Average Charge Current	Charge current to batteries
		Voltage	Total voltage of batteries
		Primary voltage	Voltage from wall socket
		Blade height	Of impeller, impacts efficiency
		Impeller Inner Diameter	Drives flow through impeller
		Impeller Outer Diameter	Drives pressure from impeller
CPM	Option Property	Washable filter (y/n)	Whether filter is washable
		Wet application (y/n)	Whether unit has wet capability
		Charge Completed Indicator (y/n)	Indicates charge completed
		Charge in Progress Indicator (y/n)	Indicates charge in progress
		Charge Management option (y/n)	Has intelligent charge controller
		Powered attachment option (y/n)	Connects external rotating brush
		Adapter storage in base (y/n)	Stores adapters in base of unit
		Noise absorber option (y/n)	Has noise absorber on impeller
		Slider switch (y/n)	Slider switch for speed control
		Clean dial (y/n)	Filter agitator to unclog filter
		Presence of Electronic Speed Control (y/n)	Speed control is electronic
		Detachable battery option (y/n)	Connects external battery pack
	Geometrical Property	Brand ID	To drive shell and handle into one
		Off unit	Anything that is not on the unit
		Front	Located in the front
		Rear	Located in the rear
	Heuristic Property	Conversion - electrical to rotational	Hits motor and axle
		Conversion - HVAC to LVAC	Hits transformer, terminals and leads
		Conversion - pressure to flow	Hits nozzle or adapter
		Dominant - hold unit	Flow of holding force
		Dominant - air flow	Flow of air through unit
		Dominant - clean filter	Flow of dust when cleaning filter
		Dominant - liquid flow	Flow of liquid for wet & dry
		Dominant - attachment power	Flow of power to attachment
		Dominant - provision electricity	Flow of power from battery to motor
	Driver Compatibility	Variance	If Technical Spec or Styling was used
		Planned change (int or ext)	If Plan Des Change or Tech Push was used

Clustering was applied to three combinations of data as shown in Table 5. More tickmarks means more information was used. Full CPM uses all the Convergence Properties. No CPM uses none, which corresponds to clustering on a normal PMM. A hybrid scenario labeled Partial CPM was introduced to show how Option Properties alone also improve dendrogram output.

Table 5. Clustering analysis was applied to three sets of data

	Data used					
Scenario	Prod. Prop.	Option Prop.	Geom. Prop.	Heur. Prop.	Driver Compat.	Module Drivers
Full CPM	✓	✓	✓	✓	✓	✓
Partial CPM	✓	✓				✓
No CPM	✓					✓

The appearance of flat subtrees in Partial CPM and No CPM signifies lack of information, and this can clearly be seen in Figure 11. It is not necessary to read the Technical Solutions to see the subtrees.

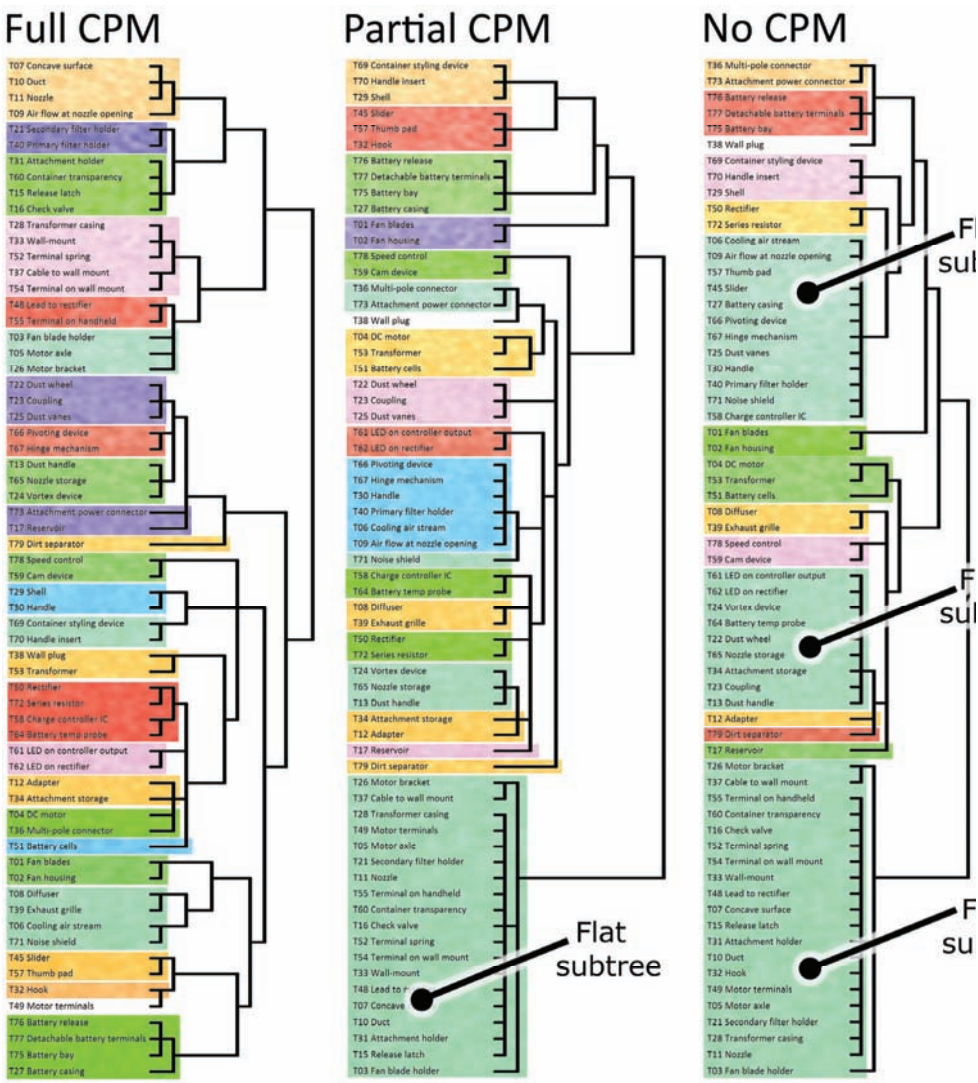

Figure 11. Dendrogram output from the three test scenarios in Table 7

Large flat subtrees *could* be module clusters, but large large flat subtrees of *unrelated* technical solutions are caused by lack of relevant scoring. The dendrogram labeled Full CPM does not show this symptom. The technical solutions of each flat subtree are listed in Table 6. The reader will be able to confirm these technical solutions indeed are not related and do not represent useful modules.

Table 6. Unrelated technical solutions in large flat subtrees

Full CPM
No flat subtrees of unrelated technical solutions
Partial CPM
Flat subtree 1
Attachment holder, Cable to wall mount, Check valve, Concave surface, Container transparency, Duct, Fan blade holder, Lead to rectifier, Motor axle, Motor bracket, Motor terminals, Nozzle, Release latch, Secondary filter holder, Terminal on handheld, Terminal on wall mount, Terminal spring, Transformer casing, Wall-mount
No CPM
Flat subtree 1
Cooling air stream, Air flow at nozzle opening, Thumb pad, Slider, Battery casing, Pivoting device, Hinge mechanism, Dust vanes, Handle, Primary filter holder, Noise shield, Charge controller IC
Flat subtree 2
LED on controller output, LED on rectifier, Vortex device, Battery temp probe, Dust wheel, Nozzle storage, Attachment storage, Coupling, Dust handle
Flat subtree 3
Hook, plus all the technical solutions in Flat subtree 1 from scenario Partial CPM

Observations from using ePMM

- The dendrograms obtained from two of the scenarios feature large flat subtrees of unrelated technical solutions, indicating these clusters are the result of insufficient information.
- The result obtained in scenario Full CPM is drastically better than from No CPM. The former corresponds to an ePMM and the latter a normal PMM without any Convergence Properties.
- With a couple of minor exceptions, the clusters made intuitive sense. This result was far better than from a typical first output of a normal PMM.
- Both the Dominant flow and Conversion-transmission heuristics were helpful, but in this particular product, the Branching-combining heuristic was not.

5 DISCUSSION

Convergence Properties in the context of other research

The proposed method presented in this paper is an attempt to improve one existing method for generation of modular architectures by including useful features of other methods while preserving the strengths of the original method upon which it is built. This approach has been used by other researchers, such as:

- Blackenfelt [2] who proposes improvements to DSM by incorporating both functional and strategic considerations
- Sellgren and Andersson [10] who incorporate interactive functions into the DSM, using a format similar to the PMM, but where the MIM is replaced by a DSM and functions take the role of properties

Conclusions

Module output varies greatly with the quality of the information provided to the clustering algorithm. The case study shows how Convergence Properties may be added to MFD in such a way that a matrix-based representation may still be used, which keeps one very important original feature intact: the possibility to apply MFD in very large projects where, for practical reason, manual module generation simply is not possible. We have seen how the addition of four proposed new property types may raise the quality of first output. The theory was tested on a product of low-to-medium complexity with about 60 technical solutions (a handheld vacuum cleaner), and yielded promising results.

Further research

More research and practical application is required to conclude whether the proposed modifications to MFD consistently improve output, in particular on more complex products with more interfaces or high innovation content. New types of Convergence Properties may be required in some cases. For example, how can Industrial Design and Manufacturing considerations be included? New types of heuristics may be required in products where no strong flows are present. That may be the case in modular storage systems (bookshelves, for example). Is there an underlying Theory of Convergence Properties, such as the Theory of Properties proposed in [11]?

REFERENCES

[1] Hölttä-Otto, K. Modular Product Platform Design, 2005 (Doctoral Dissertation, Dept. of Mechanical Engineering, Helsinki University of Technology, Espoo).

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

[3] Ulrich, K. T. and Eppinger, S. D. Product Design and Development, fourth edition, 2008 (McGraw-Hill Education Asia, Singapore).

[4] Erixon, G. Modular Function Deployment - A Method for Product Modularisation, 1998 (Doctoral Thesis, Dept. of Manufacturing Systems, Royal Institute of Technology, Stockholm).

[5] Akao, Y. and Mizuno, S. QFD: The Customer-Driven Approach to Quality Planning and Development, 1994 (Asian Productivity Organization, Tokyo).

[6] Nilsson, P. and Erixon, G. The Chart of Modular Function Deployment. In *4th Workshop on Product Structuring*, Delft 1998, (Delft University of Technology, The Netherlands).

[7] Stake, R. On conceptual development of modular products, 2000 (Doctoral Thesis, Dept. of Production Engineering, Royal Institute of Technology, Stockholm).

[8] Stone, R. B., Wood, K. L., and Crawford, R. H. A Heuristic Method for Identifying Modules for Product Architectures. In *Design Studies*, vol. 21, no. 1, pp. 5-31, 2000.

[9] Zamirowski, E. J. and Otto, K. N. Identifying Product Portfolio Architecture Modularity Using Function and Variety Heuristics. In *Proceedings of the 1999 ASME Design Engineering Technical Conferences, DETC99/DTM-8760*, Las Vegas, September 1999, pp. 187-197 (The American Society of Mechanical Engineers, New York)

[10] Sellgren, U. and Andersson, S. The Concept of Functional Surfaces as Carriers of Interactive Properties. In *International Conference on Engineering Design (ICED 05)*, Melbourne, August 2005.

[11] Hubka, V. and Eder, W. E. Design Science, 1996 (Springer-Verlag London Limited).

Contact: Fredrik Börjesson, Department of Machine Design, Brinellvägen 85, Royal Institute of Technology (KTH), SE-100 44 Stockholm, Sweden
E-mail fborjesson0524@hotmail.com

Fredrik Börjesson is Master of Science Electrical Engineering and PhD student at the Royal Institute of Technology (KTH) in Stockholm, Sweden. He is also the VP of Technology in the Sweden-based consulting firm Modular Management AB.

The author wishes to acknowledge the kind support of Dr. Mark W. Lange, Dr. Gunnar Erixon, Professor Ulf Olofsson, and Associate Professor Ulf Sellgren.