Henri Savolainen

DEFINING AND GOVERNING DESIGN KNOWLEDGE MODULES AND INTERFACES IN VALVE TOPWORKS ASSEMBLY

Faculty of Engineering and Natural Sciences
Master of Science Thesis
September 2019

ABSTRACT

Henri Savolainen: Defining and Governing Design Knowledge Modules and Interfaces in Valve Topworks Assembly
Master of Science Thesis
Tampere University
Master’s Degree Programme in Mechanical Engineering
September 2019

Modularity has long played a major role in industrial companies. Modularity has been used in many ways in products and successful modularization has been found to provide a competitive advantage. However, the idea, at the heart of modularity, that weakening linkages between modules allowing them to be managed as independent entities has not been exploited in design knowledge governance.

Metso Flow Control has implemented KBE-software Rulestream to support valve product development. In Rulestream, design information is compiled into design rules that are used by the software to create 3D models and drawings. It has been decided to extend the use of the software to include the creation of a valve topworks assembly.

This work defines the entities in which the design rules will be governed. The design rules need to be divided into manageable modules so that the design rules can be managed in the future. The thesis investigates how the valve topworks assembly can be divided into modules that support knowledge governance. The interfaces between the modules are critical to the operation of the entire system.

The research method used is case study research, which is based on a literature review. The literature review examines modularity, modules, interfaces and modularization methods. Also some literature related to data governance will be reviewed. Based on the literature review, a method that is applicable to the modularization of the valve topworks assembly is selected.

The tool chosen for modularization of the valve topworks assembly is the Design Structure Matrix. The DSM is a tool that has been used extensively in product modularization. DSM also provides valuable information on the system. Several matrices are produced and the most value generating solution is selected. Based on the matrix, the design modules and their interfaces are determined. Design modules need to support knowledge management, so they need to mirror the existing organizational structure. Modules are independent entities that can be modified without affecting other modules. Interdependencies between modules are manifested in module interfaces. Design knowledge that affects more than one module is the interface between the modules. Governing this kind of information is very important. Module governance requires specification of the owners of the modules and interfaces. The owner is responsible for what changes can be made to the modules and interfaces. The module partitioning and ownership defined in the thesis support the maintenance of design knowledge.

Based on the modules defined in the thesis, a CAD skeleton model of valve topworks assembly was created. The skeleton model is built from defined modules and it documents the physical interfaces needed in connecting the modules.

Keywords: Modularity, Design Structure Matrix, Design knowledge governance

TIIVISTELMÄ

Henri Savolainen: Venttiiliyhdistelmän suunnittelutietomoduuleiden ja rajapintojen määritys ja hallinta
Diplomityö
Tampereen yliopisto
Konetekniikan diplomi-insinöörin tutkinto-ohjelma
Syyskuu 2019

Modulaarisuus on ollut suuressa roolissa teollisuusyrityksissä jo pitkään. Modulaarisuutta on käytetty tuotteissa monin tavoin ja onnistuneen moduloinnin on todettu tuottavan kilpailuetua. Kuitenkin modulaarisuuden ytimessä olevaa ajatusta siitä, että moduulien välisten sidosten heikentäminen mahdollistaa niiden hallinnan itsenäisinä kokonaisuuksina ei ole hyödynnetty suunnittelutiedon hallinnassa.

Metso Flow Control on käyttöönottanut KBE-ohjelmisto Rulestreamin venttiilien tuotekehityksen tueksi. Rulestreamissa suunnittelutieto kootaan suunnittelusäännöiksi, joiden pohjalta ohjelmisto luo 3D-malleja ja piirustuksia. Ohjelmiston käyttö on päätetty laajentaa kattamaan myös venttiiliyhdistelmän kokoonpanon luominen.

Tässä työssä määritellään millaisina kokonaisuuksina suunnittelusäännöt tulisi hallita. Suunnittelusäännöt halutaan jakaa hallittaviin moduuleihin, jotta suunnittelusäännöstön ylläpito on mahdollista tulevaisuudessa. Työssä tutkitaan, miten venttiiliyhdistelmä voidaan jakaa moduuleihin, jotka tukevat tiedonhallintaa. Moduuleiden välillä ovat rajapinnat, jotka ovat kriittisiä koko systeemin toiminnan kannalta

Tutkimusmenetelmänä käytetään tapaustutkimusta, jonka pohjana toimii kirjallisuuskatsaus. Kirjallisuuskatsauksessa perehdytään modulaarisuuteen, moduuleihin, rajapintoihin ja modulointimenetelmiin. Myös kirjallisuutta tiedonhallinnasta esitellään. Kirjallisuuskatsauksen pohjalta valitaan menetelmä, jota sovelletaan venttiiliyhdistelmän modulointiin.

Venttiiliyhdistelmän modulointiin valittu työkalu on Design Structure Matrix. DSM-työkalua on käytetty paljon tuotteiden moduloinnissa. DSM antaa myös arvokasta tietoa systeemistä. Matriiseja tehdään useita ja niistä valitaan eniten arvoa tuottava ratkaisu. Matriisin pohjalta määritetään suunnittelumoduulit ja niiden rajapinnat. Suunnittelumoduulien on tuettava tiedonhallintaa, joten niiden on peilattava olemassaolevaa organisaatiorakennetta**.** Moduulit ovat itsenäisiä kokonaisuuksia joiden sisällä voidaan tehdä muutoksia vaikuttamatta muihin moduuleihin. Moduuleiden väliset riippuvuudet ilmenevät moduuleiden rajapinnoissa. Suunnittelutieto, joka vaikuttaa useampaan moduuliin, on rajapinta moduuleiden välillä. Tällaisen tiedon hallinta on erittäin tärkeää. Moduuleiden hallinnan takia moduuleille ja rajapinnoille on määritettävä omistajat. Omistaja vastaa siitä, mitä muutoksia moduuleihin ja rajapintoihin voidaan tehdä.

Työssä määrittetty moduulijako ja omistajuus tukevat suunnittelutiedon ylläpitoa. Työssä määritettyjen moduuleiden pohjalta luotiin CAD-luurankomalli venttiiliyhdistelmästä. Luurankomalli rakentuu määritetyistä moduuleista ja siihen on dokumentoitu tarvittavat fyysiset rajapinnat moduuleiden yhteenliittämiseen.

Avainsanat: Modulaarisuus, Design Structure Matrix, Suunnittelutiedon hallinta

PREFACE

This Master’s Thesis was done for Metso Flow Control Oy. This research was conducted between February and July in 2019. The topic was not familiar to me before starting this thesis. The topic was interesting, so I decided to go for it.

I would like to thank everyone in Metso Flow Control who has participated in this research. Big thanks go to my supervisor Ville Hiltunen for guiding me through this process. Secondly, I would like to thank University Lecturer Timo Lehtonen who gave me valuable tips at the beginning of the research.

Biggest thanks go to my family and friends who have pushed me through university. Especially I want to thank my girlfriend Ellinoora who has offered me her unwavering support and has listened to my complaints and worries throughout the process.

Vantaa, 3 September 2019

Henri Savolainen

CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

AICB	Architecture Interface Control Board
CAD	Computer-Aided Design
DFX	Design for X
DPM	Design Property Matrix
DSM	Design Structure Matrix
DSS	Design Structure System
ECN	Engineering Change Notice
ECP	Engineering Change Proposal
ETO	Engineering to Order
KBE	Knowledge-Based Engineering
MFC	Metso Flow Control
MFD	Modular Function Deployment
MIM	Module Indication Matrix
PMM	Product Management Map
QFD	Quality Function Deployment

1. INTRODUCTION

Modularity is a widely researched method for organizing complex systems into blocks that are easier to manage. The system that will be organized into modules can be almost anything such as machine or software. Many industrial companies use modularity to partition the architecture of products into manageable blocks. These modules often have an appointed owner that is responsible for the module.

The main idea behind modularity is dividing the system into manageable blocks and weakening the couplings between the blocks. This enables the modules to be designed separately. This idea can be applied to design knowledge management.

Metso has done a design automation project which aims to speed up product development and reduce manual work. Design automation is executed by using Rulestream-software. This enables the design knowledge to be captured into well documented design rules. The design rules are divided into general rules and product specific rules. It is crucial to define processes and practices for data governance because the design rules will be used globally and changes in them affect every component where the changed rule is used. The design knowledge of valves has already been collected into design rules. The next step is to expand the design automation to include the design of other components in valve topworks assembly.

This research aims to determine how this kind of design knowledge should be structured and governed. The design knowledge of valve topworks assembly will be divided into modules and the interfaces between the design modules will be discovered. The governance model for the defined modules and interfaces will be created. Assembly skeleton for the automated CAD-assembly of valve topworks assembly will also be created.

2. THE OBJECTIVES OF THE RESEARCH AND RESTRICTIONS

This chapter describes the objectives and restrictions of the research and the research questions. Also the structure of the thesis is presented.

2.1 Objectives and research questions

The goal of this research is to find a way to partition design knowledge into manageable modules. Modularity of design knowledge is not a widely researched topic. Metso has implemented design automation for valve design. The design automation will be expanded to valve topworks assembly's other components. This will allow automating the CAD-assembly of valve topworks assembly. The CAD-assembly will also need an assembly skeleton for it to work.

The scope of this research is to concentrate on the assembly level. Valve, actuator, positioner/limit switch, instrumentation and mounting sets are the components that will be studied. The focus is on the mechanical structure and mechanical design dependencies between the components.

The research questions were created based on the research problem:

Research question 1: How to define design entities that support the knowledge governance?

Research question 2: What are the interfaces between the modules and how they can be discovered?

Research question 3: How should the design entities and interfaces be governed?

2.2 Restrictions of the research

This thesis focuses on finding the design modules at the level of valve topworks assembly. The design of components in valve topworks assembly could also be divided into modules but that is not in the scope of this research. The focus is solely on the assembly level of valve topworks assembly and the components are only part of the system. The linear actuators and valves won't be included in this research because the structure is different from quarter turn actuators and valves.

The design knowledge that is at the focus of this research is the general rules that affect the design of all product families. The product specific rules affect only specific product families. Determining the ownership of product specific rules is not at the scope of this research.

2.3 Structure of the thesis

Chapter three of this thesis reviews the literature relevant for the scope of this study. Relevant theory related to modularity and modularization methods will be presented. The literature related to governing data will also be presented.

Chapter four presents the research method and process. First the nature of the research and literature related to that will be presented. The chosen method for modularization of design knowledge will be introduced. Also, a process for collecting and refining the information relevant for this research is presented.

Chapter five focuses on Metso, its products and the Metso's need for this thesis. The valve topworks assembly's components are introduced and the design automation is briefly explained.

Chapter six presents the use of design structure matrix in this thesis. The process of collecting information for the matrix and different variations of matrices will be presented.

Chapter seven presents the results of this thesis. The design knowledge modules defined with the use of DSM will be presented. The governance model for the design knowledge modules will be introduced. Finally, the assembly skeleton that is based on the design knowledge modules is briefly presented.

Chapter eight answers to the research questions presented in the beginning of this thesis. The research process will be evaluated. This chapter will also present suggestions for future research. Chapter nine summarizes the thesis.

3. LITERATURE REVIEW

To support the research questions, the theory chapter of this work will review literature related to the context of the study. The starting point for the literature review was modularity as a concept and modularization methods. Also literature of governing data was examined.

3.1 Theory of Domains

Theory of Domains is a theory that suggests that design of system consists of four domains. The domain is the level of abstraction of the system. Domains can be considered as viewpoints to the product. Domain theory explains the chain of design that leads to the physical product. Technical systems can be broken down into these domains. The domains are process system, effect system, organ system and part system. Process system describes the transformation that the system makes. Effect system consists of functions of the system. These are often defined from the viewpoint of salesperson. Organ system consists of solutions that enable the functionality of the technical system. This is usually the designer's viewpoint. Part system consists of parts of the system. Manufacturing happens at this level. (Andreasen 1980) The domains are presented in figure 1.

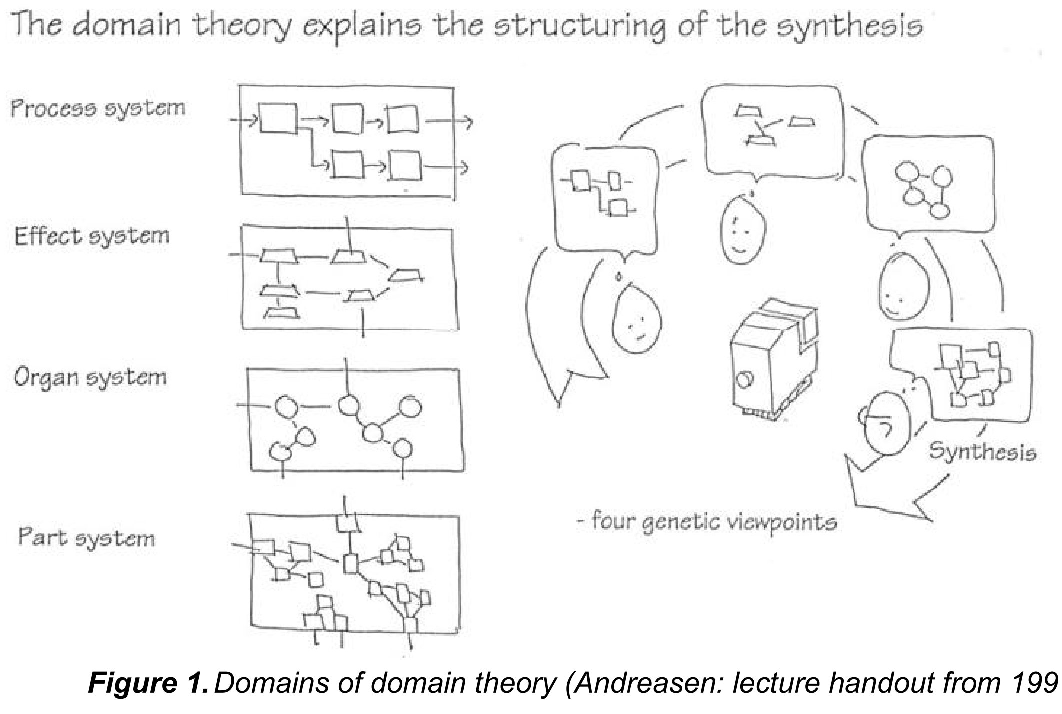

The domain theory explains the structuring of the synthesis
Process system
Effect system
Organ system
Part system
Synthesis
- four genetic viewpoints

Figure 1. *Domains of domain theory (Andreasen: lecture handout from 1997)*

Using the function domain in the design of modular architecture has been researched widely. According to Ulrich and Eppinger (2012) defining of the functions of the product and splitting up the overall functions into sub-functions is key step in modularization. Fujimoto (2007) states that subfunctions affect the division of modules. Subfunctions define the modules characteristics. The connection between functional design and structural design is presented in figure 2.

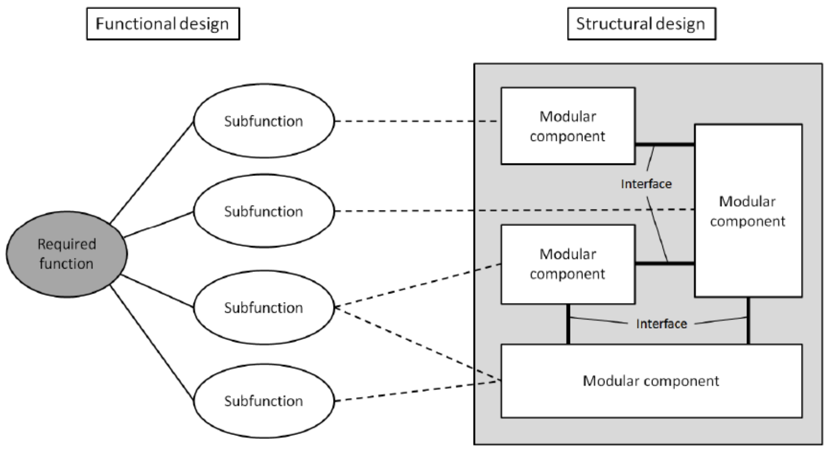


Figure 2. *Correlation between functions and modules (Fujimoto 2007)*

According to Fujimoto (2007) each module in structural design corresponds to a specific function. Between these modules are the interfaces. This leads to connection between Theory of Domains and modularity.

3.2 Modularity

Modularity is a concept that has proved useful in many fields that deal with complex systems. Modularity can be thought as a strategy for organizing complex products and processes efficiently (Baldwin & Clark 1997). According to Andreasen (2011) complexity in operations of a company can be reduced by defining a modular architecture with module and interface definitions. Modularity has been adapted widely not only in physical products but also in non-physical types of products such as in software industry.

In literature modularity is often described as a relative property of product which increases when the functional structure and physical structure become similar. This minimizes the dependencies between the various elements of the system. (Ulrich & Tung 1991)

Ulrich and Tung (1991) define that:

"*Modularity arises from the way a product is physically divided into independent components*."

The dependencies between the elements of the system that modularity aims to minimize can be many things. Depending on what are drivers for modularity. In literature these dependencies are often called couplings (Sanchez & Mahoney 1996, Borjesson & Hölttä-Otto 2013). Sanchez & Mahoney (1996) state that components can be loosely coupled or tightly coupled. The strength of the coupling depends on the extent to which a change in one element requires change in the other element. This kind of high degree of independence can be created by standardizing the elements interfaces.

Standardization means replacing several components from a system with a single component which can perform the same functions. (Perera et al. 1999) According to Pakkanen (2015, p. 53) standardization enables modularization.

Modularity can be achieved by partitioning information into visible design rules and hidden parameters (Baldwin & Clark 1997). Visible rules define the product and affect the design decisions. Visible rules are divided to three categories:

- **Architecture**, which specifies what modules will be part of the system and what their functions will be.
- **Interfaces** that describe in detail how the modules will interact, including how they will fit together, connect and communicate.
- **Standards** for testing module's conformity to the design rules and for performance relative to another. (Baldwin & Clark 1997)

Hidden design parameters are decisions that do not affect the design beyond the local module. Hidden parameters do not have to be communicated to anyone beyond the module design team. (Baldwin & Clark 1997)

Lehtonen (2007 p. 88) presents two approaches to modularity: M-modularity that means modularity aiming at configuration and modularity related to the life cycle of the product. The letter M in M-modularity comes from finnish word for configuration *(muuntelu).* Life cycle modularity is often not visible in the end product.

In Modularity related to the life cycle of the product modularity is related to three type categories. The categories are:

- Modularity based on reasons of manufacture
- Modularity based on reasons of maintenance
- Modularity based on logistical reasons

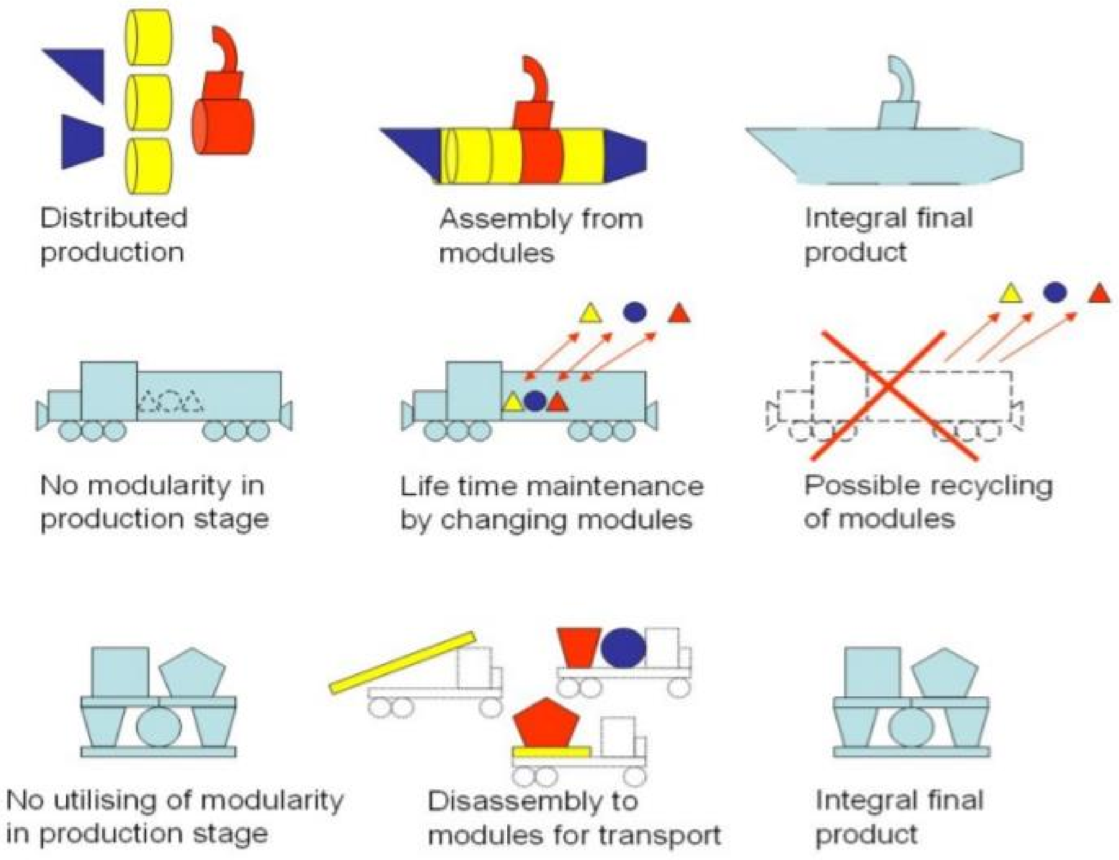

Distributed production
Assembly from modules
Integral final product
No modularity in production stage
Life time maintenance by changing modules
Possible recycling of modules
No utilising of modularity in production stage
Disassembly to modules for transport
Integral final product

Figure 3. *The categories of modularity related to the life cycle of the product (Lehtonen 200, p. 90)*

Figure 3 presents the above-mentioned categories of modularity related to the life cycle of the product. The modules are highlighted in colors. In modularity based on reasons of manufacturing the product is produced in distributed modules and then moved to the assembly site. Modularity is utilized only in the production phase and the product is assembled using the produced modules. Modularity is not visible in the integral final product. Example of this is the manufacturing of a submarine. In modularity based on reasons of maintenance the modules are changed as part of maintenance. These modules can be potentially recycled at the end of the life cycle. Modularity is not present in the production stage. Example of this is a locomotive. In modularity based in logistical reasons the integral product is disassembled into modules for transport and then assembled into integral product. Modularity is not utilized in production stage or in integral final product. Life cycle modularity does not involve configuration. (Lehtonen 2007, pp. 89-90)

3.2.1 Modular system

Lehtonen (2007 p. 32) states that Karl-Heinz Borowski's work can be considered fundamental from the viewpoint of modularity. Borowski (1961) presents in his Das Baukastensystem in der Technik book a product family system based on constructional elements (Ger. Baustein). A constructional element (Ger. Bauskasten) is an element on the level to be examined that consists of small constructional elements (Ger. Baustein). This can be considered the base for modular system.

Lehtonen (2007 p. 88) defines a modular system in as follows:

"A modular system is a system consisting of blocks which involves the interchangeability of the blocks."

In modularity these blocks are called modules. The interchangeability can be achieved by weakening the links between the blocks. According to Baldwin and Clark (1997) a modular system is composed of units (or modules) that are designed independently but still function as an integrated whole.

3.2.2 Module

Modules are the building blocks in a modular system. According to Andreasen (2011, p. 302) a module is a product's independent entity that fulfils requirements from the viewpoint of functions or organs. The weak dependencies between different modules allow them to be independent. This allows for modules to be developed separately.

Lehtonen (2007 p. 88) defines a module in as follows:

"*A block (any assembly of the product or part of the system) is a module if it has an assigned interface and it is a part of a modular system*."

Modules have **interfaces** between them because modules need to connect to each other. By defining and standardizing these interfaces the benefits for modularity can be achieved**.**

3.2.3 Interface

Interfaces exist between the modules. Interfaces connect modules and allow for component swapping. The modules can be interchangeable only if they have compatible interfaces (Miller & Elgård 1998, pp. 10-14).

According to Miller & Elgård (1998, p. 14) the interfaces are boundaries between the modules. The interfaces can be divided into different types. The types are functional, mechanical and electrical. The interaction between the modules over the interfaces can be divided into four different types. The types of interaction are energy, information, material and spatial.

Parslov and Mortensen (2015) state that interfaces in modular product can be divided to two types. The A-type – interfaces are interfaces between modules where there is high variation. The definition of these interfaces is crucial. The B-type interfaces are interfaces between components such as physical contact points with transfer of work, current and heat. These interfaces are not as strategically important as the A-type- interfaces.

According to Ullman (1992) connections between components require most design effort. Therefore, it is important to focus on the interfaces in product development. Specification of the interfaces is a key part of the product architecture and modularization (Ulrich 1995).

According to Sanchez & Mahoney (1996) a firm must have access to advanced architectural knowledge about relevant components and their interactions to fully specify component interfaces in modular product architecture. Standardizing components interfaces can be used to form loose couplings between components (Sanchez & Mahoney 1996).

3.3 Modular Product architecture

Garud and Kumaraswamy (1993) describe modular product architecture as a special form of product design that uses standardized interfaces between components to create a flexible product architecture. Modular product design comes from standardized interfaces that allow specific range of variation in components. Standardizing allows the flexibility that manifests itself in swapping components without the need for redesign.

Modular product architecture consists of modular components. According to Sanchez & Mahoney (1996) modular product architecture defines the range of variation for the interface characteristics of modular components.

Module systems can be divided into module systems and mixed systems. Modular systems can be made from equal modules, different types of modules and combinations of modules and custom parts. Module system consisting of equal modules is constructed from only similar modules, only different modules or combination of similar and different modules. Module system consisting of different types and sizes of modules is constructed from small modules that add functionalities to the system and bigger modules that work as a platform for the smaller modules. The small modules can be equipment modules, accessory modules or joining modules. Mixed systems consist of modules and custom integral components. Modular system cannot be considered as a modular system if the modularity is based only on modular components joining the integral components. (Brankamp & Herrmann cited in Lehtonen 2007 p. 40) The different types of module systems are illustrated in figure 4.

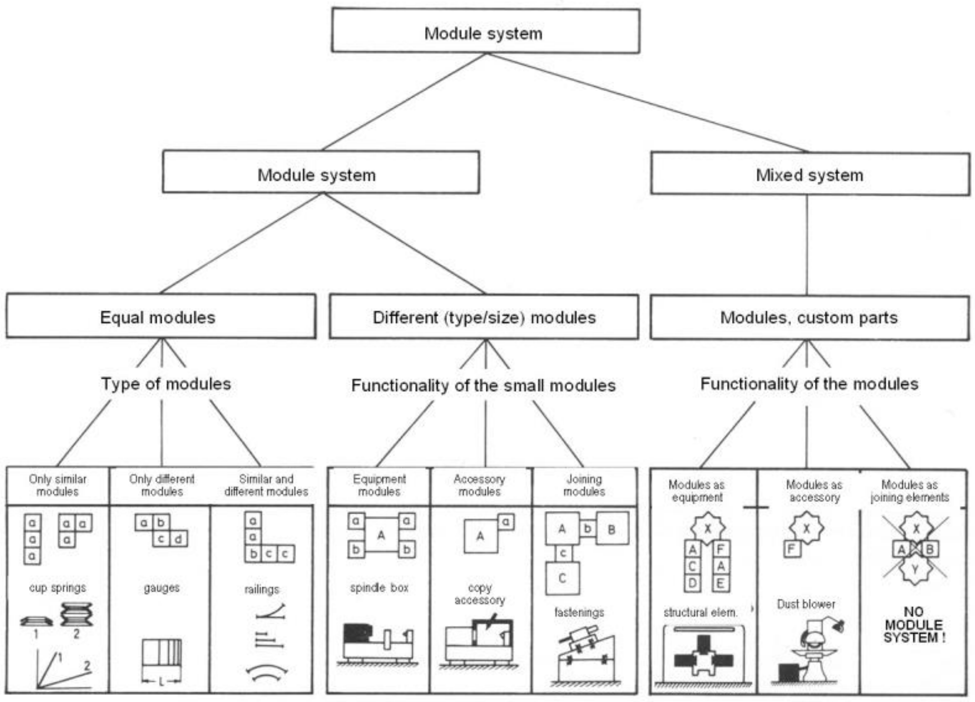


__Figure 4.__ Different types of module systems (Lehtonen 2007, p. 40)

Pakkanen (2015, p. 10) suggests that modular product family design involves five key design elements. The key design elements are:

- Partitioning logic
- A set of modules
- Interfaces (standardized)
- Architecture
- Configuration knowledge

Partitioning logic defines the viewpoints that affect product structuring decisions from a business and customer perspective. Set of modules defines what modules are included in product variants of a product family. Standardized interfaces enable efficient defining of product variants in the order/sales-delivery process. Architecture describes the relations between modules and interfaces. Configuration knowledge links the modules to customer needs. (Pakkanen 2015, p. 10)

3.4 Types of modularity

Pine (1993, according to Pakkanen 2015, p. 54) recognizes six types of modularity. The types are presented in figure 5. Pakkanen (2015, p. 54) describes the types as follows:

- **Component-sharing modularity** uses same components in multiple products.
- **Component-swapping modularity** complements component-sharing modularity. Components are paired with the same basic product and as many products as there are components to be swapped can be created.
- **Cut-to-fit modularity** considers one or more components which are continually variable within pre-set or practical limits.
- **Mix modularity** considers mixing components together in a way that something different is created.
- **Bus modularity** uses standard structure in which a number of different components can be attached.
- **Sectional modularity** allows the configuration of types of components which have standard interfaces in arbitrary ways. This type of interchangeability allows the greatest degree of variety and customisation and enables re-configurability, but it is the most difficult to achieve

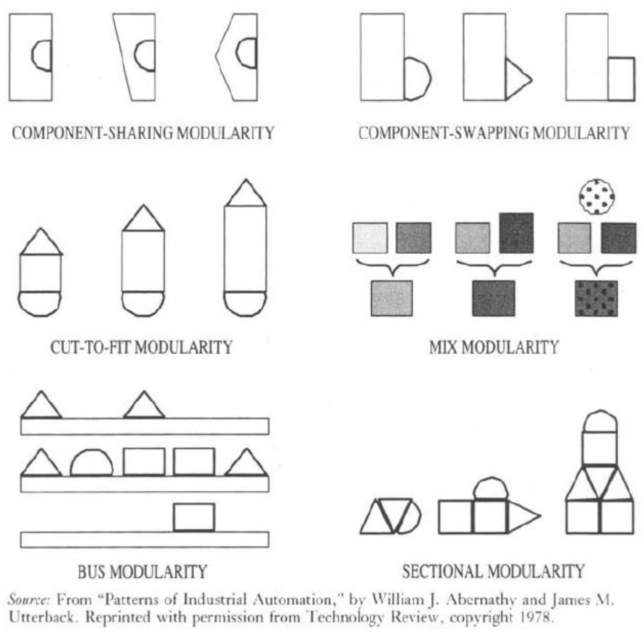

COMPONENT-SHARING MODULARITY
COMPONENT-SWAPPING MODULARITY
CUT-TO-FIT MODULARITY
MIX MODULARITY
BUS MODULARITY
SECTIONAL MODULARITY
Source: From "Patterns of Industrial Automation," by William J. Abernathy and James M. Utterback. Reprinted with permission from Technology Review, copyright 1978.

Figure 5. *Types of modularity (Pine 1993)*

Additionally, two more types of modularity have been presented in literature. Miller & Elgård (1998) introduced stack modularity and Danish doctoral students introduced On-off modularity. These two types are presented in figure 6. In stack modularity paramet-rical configuration is implemented by multiplying the number of modules. In On-off mod-ularity a place is reserved for module even if it is not chosen.

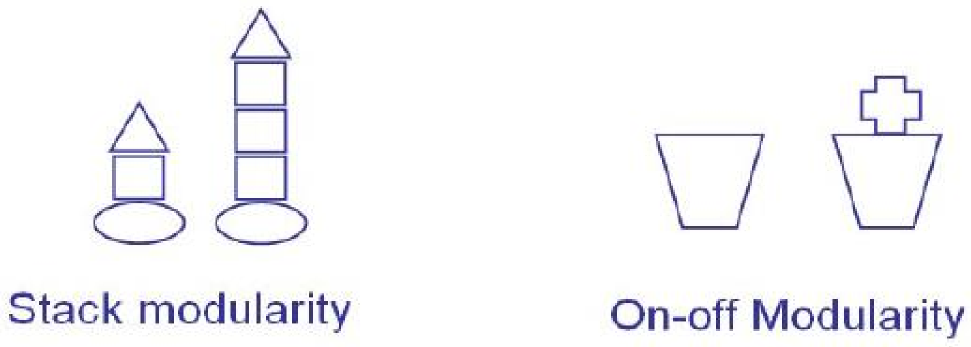


Figure 6. *Stack and On-off modularity (Lehtonen 2007, p. 49)*

3.5 Development of modular product architecture

Figure 7 presents the level of modularity's effect on implementation of the modularity and the benefits and goals of each level. Standardization is the base of all modularity. The lowest level of modularity is assembly-based modularity. This is type of modularity is most common. In this level modular division is often performed from the viewpoint of production and maintenance. (Lehtonen 2007 p. 92)

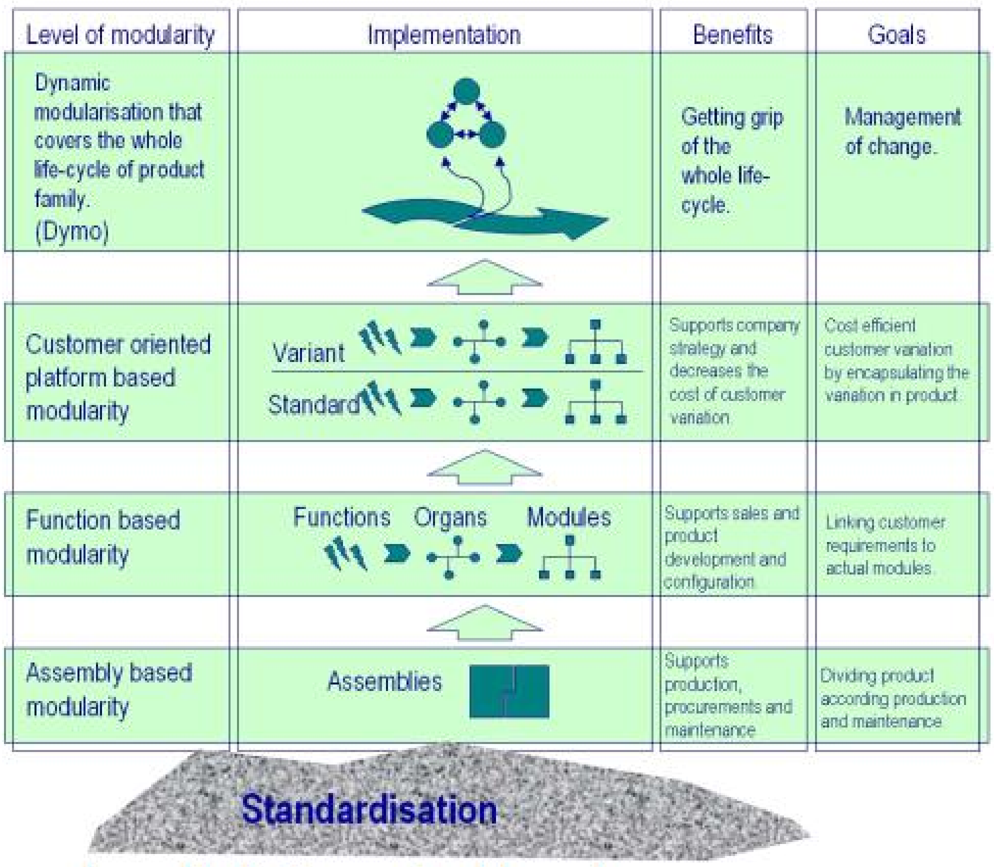

Level of modularity
Implementation
Benefits
Goals
Dynamic modularisation that covers the whole life-cycle of product family. (Dymo)
Getting grip of the whole life-cycle.
Management of change.
Customer oriented platform based modularity
Variant
Standard
Supports company strategy and decreases the cost of customer variation
Cost efficient customer variation by encapsulating the variation in product
Function based modularity
Functions
Organs
Modules
Supports sales and product development and configuration
Linking customer requirements to actual modules.
Assembly based modularity
Assemblies
Supports production, procurements and maintenance
Dividing product according production and maintenance
Standardisation

Figure 7. *Development of modular product structures (Lehtonen 2007, p. 92)*

The second level of modularity, function-based modularity focuses on the functions of the product. These divide into organs and finally to modules. This could potentially support sales, product development and configuration. Designing the modules according to the functions links the customer requirements to actual modules. (Lehtonen 2007 p. 93)

The third level of modularity is the customer-oriented platform-based modularity. Platform is based on functional modularity. This differs from the second level in that the product is divided into variable elements and a stabile standard element. This kind of modularity supports company strategy and decreases customer variation. Biggest benefit is the potential for cost efficient customer variation. (Lehtonen 2007 p. 93)

The last highest level of modularity presented in figure 7 is the dynamic modularisation. Dynamic modularisation includes the life cycle of the product in the product platform. This enables the management of change in the product. (Lehtonen 2007 pp. 92-94)

3.6 Modularization methods

There are many modularity methods for creating modular product architectures. Module definition methods can be divided into two categories based on the data format: matrix-based approaches and graphical function network-based approaches. Methods can also be categorized into coupling- and similarity-based approaches. (Borjesson & Hölttä-Otto 2013)

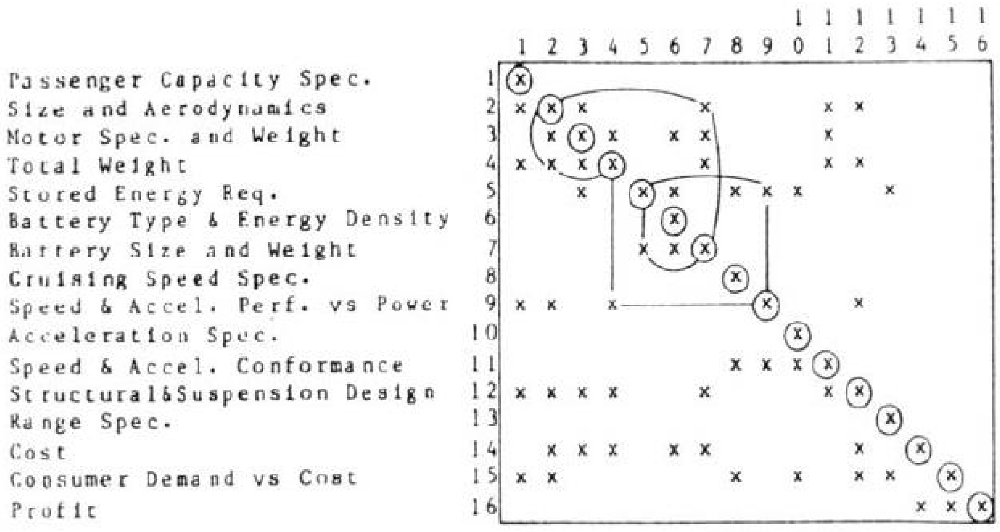
Coupling-based approaches aim to cluster elements into modules by maximizing the coupling or connectivity within the modules and minimizing the coupling between the modules. In Similarity-based approaches define the modules are defined based on similarity and dissimilarity. The elements are compared according to strategic drivers and product properties. Often elements that are different from the rest but similar to each other are isolated into modules in this approach. (Borjesson & Hölttä-Otto 2013)

According to Hölttä and Salonen (2003) there are only three systematic modularity methods: Function Structure Heuristic method, clustering a Design Structure Matrix and Modular Function Deployment. Design Structure Matrix is usually best suited method for defining modules within a single product's architecture.

3.6.1 Dependency matrices

As it has been stated before. Independence of the elements that form system is a key factor in modularity. The independence can be evaluated efficiently by using a dependency matrix. Steward (1981) introduced a theory for design project management called Design Structure System (DSS).

DSS is based on a square matrix in which the subtasks of design are entered on the horizontal rows in order of their execution. The same tasks are entered on the vertical columns in the same order. Each task in the columns is examined and it is discovered whether the tasks are dependent of each other. (Steward 1981)



		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Passenger Capacity Spec.	1	ⓧ															
Size and Aerodynamics	2	x	ⓧ	x				x				x	x				
Motor Spec. and Weight	3		x	ⓧ	x		x	x				x					
Total Weight	4	x	x	x	ⓧ			x				x	x				
Stored Energy Req.	5			x		ⓧ	x		x	x	x			x			
Battery Type & Energy Density	6						ⓧ										
Battery Size and Weight	7					x	x	ⓧ									
Cruising Speed Spec.	8								ⓧ								
Speed & Accel. Perf. vs Power	9	x	x		x					ⓧ			x				
Acceleration Spec.	10										ⓧ						
Speed & Accel. Conformance	11								x	x	x	ⓧ					
Structural&Suspension Design	12	x	x	x	x			x				x	ⓧ				
Range Spec.	13													ⓧ			
Cost	14		x	x	x		x	x					x		ⓧ		
Consumer Demand vs Cost	15	x	x						x		x		x	x		ⓧ	
Profit	16														x	x	ⓧ

Figure 8. *Design Structure System (Steward 1981)*

A matrix filled as illustrated in Figure 8 is a directional matrix: each relation is marked with a x. The goal of DSS is to eliminate the relations above the diagonal line and form a bottom triangle. The relations above the diagonal line are problematic because they represent iterations back to the previous task. (Steward 1981) Optimizing the matrix can be done heuristically or mathematically.

When the matrix arrangement nears the optimum, the relations causing iterations form limited task groups that are called clusters. The tasks in a cluster are the smallest possible number of tasks that must be simultaneously examined when design cannot be performed iteratively based on estimations. Dividing project tasks into independent clusters is called partitioning. (Lehtonen 2007, p.51) The clustered DSS is presented in figure 9.

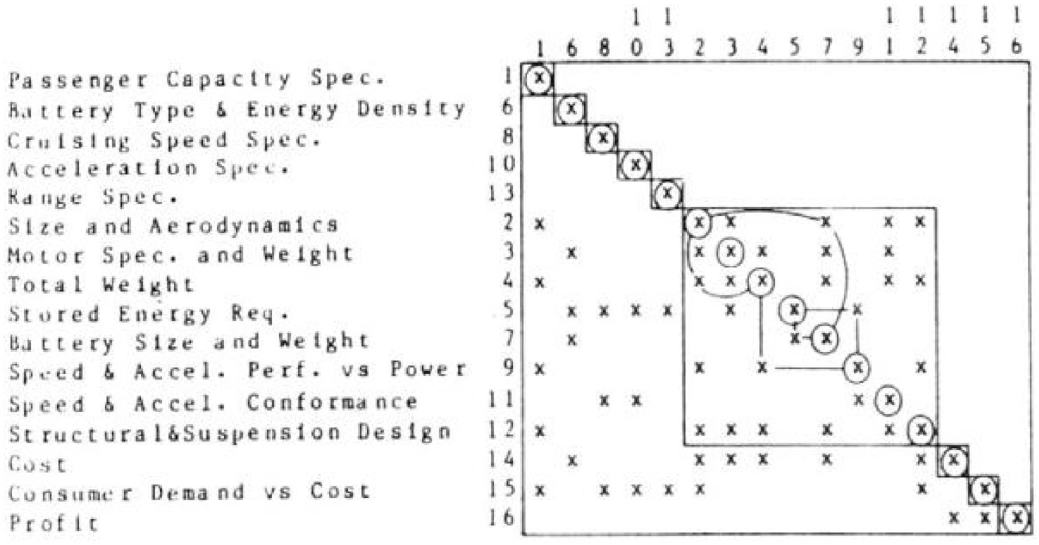


Figure 9. *Clustered DSS (Steward 1981)*

Malmqvist (2002) summarises the elements that can be included in matrix-based modelling methods. The elements can be properties, functions, subsystems/organs/design parameters/features, components, life-cycle systems/processes or product level alternatives or variants. There are many different matrix tools. Figure 10 presents the different matrix-based product modelling methods.

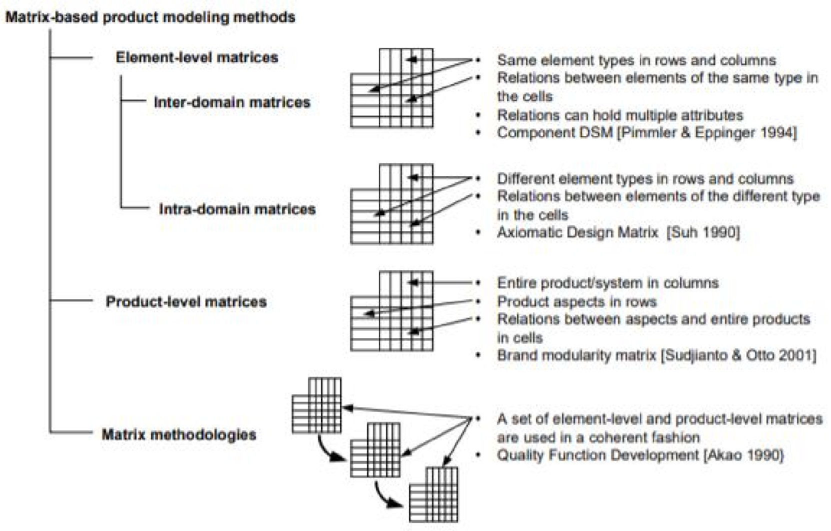

Matrix-based product modeling methods
Element-level matrices
Inter-domain matrices
Intra-domain matrices
Product-level matrices
Matrix methodologies
Same element types in rows and columns
Relations between elements of the same type in the cells
Relations can hold multiple attributes
Component DSM [Pimmler & Eppinger 1994]
Different element types in rows and columns
Relations between elements of the different type in the cells
Axiomatic Design Matrix [Suh 1990]
Entire product/system in columns
Product aspects in rows
Relations between aspects and entire products in cells
Brand modularity matrix [Sudjianto & Otto 2001]
A set of element-level and product-level matrices are used in a coherent fashion
Quality Function Development [Akao 1990]

Figure 10. *Matrix-based modelling methods (Malmqvist 2002)*

Malmqvist (2002) divides the methods into element-level matrices, product level matrices and matrix methodologies. Element level matrices can be divided into inter-domain matrices and intra domain-matrices. The inter-domain matrices such as component DSM by Pimmler and Eppinger (1994) use same element types in rows and columns. Intra-domain types use different types of elements in rows and columns. Suh (1990) presented axiomatic design matrix that is an example of intra-domain matrix. Product-level matrices put entire products into the columns and product aspects into rows. Example of product-level matrix is the brand modularity matrix by Sudjianto and Otto (2001). Matrix methodologies use different types of element-level and product-level matrices in coherent fashion. This is applied in quality function development by Akao (1990).

According to Malmqvist (2002) there are seven methods of analysis for matrices. The methods are:

- Clustering, in which the elements are organized in a way that strong internal relations are gathered into clusters with weak external relations
- Partitioning, in which the iterations in the process are minimized
- Coverage, in which the allocated functions that are not realized are detected
- Index computation, in which indices are computed to produce deductions
- Interaction focuses on content of individual relations and eliminating harmful effects
- Change propagation, in which the impact of change proposal can be evaluated
- Alignment, in which two related matrices are compared to highlight differences that might lead difficulties in managing interfaces

According to Lehtonen (2007 p. 55) clustering, partitioning, index computation and alignment can be considered in producing a modular structure. In analysing a modular structure all methods are applicable.

3.6.2 Design structure matrix

Widely known example of dependency matrices is the research on the composition of engine design teams. McCord and Eppinger (1993) call their matrix a Design Structure Matrix (DSM). The engine is an integral product in the sense that it contains a number of strong dependencies, which means that partitioning was not entirely successful. However, two clusters could share some elements because the goal was to compose optimal teams. The formed teams performed extremely well so this research speaks on behalf of using DSM in partitioning and visualizing these kinds of relations.

McCord and Eppinger (1993) used three types of markings to describe the strength of the dependencies. The dependencies were differentiated into high, average and low. The goal was that strong dependencies would be placed into same cluster. In their use of dependency matrix, the matrix becomes symmetric because dependencies go both ways. The starting point of the study and the clustered results are presented in figure 11.

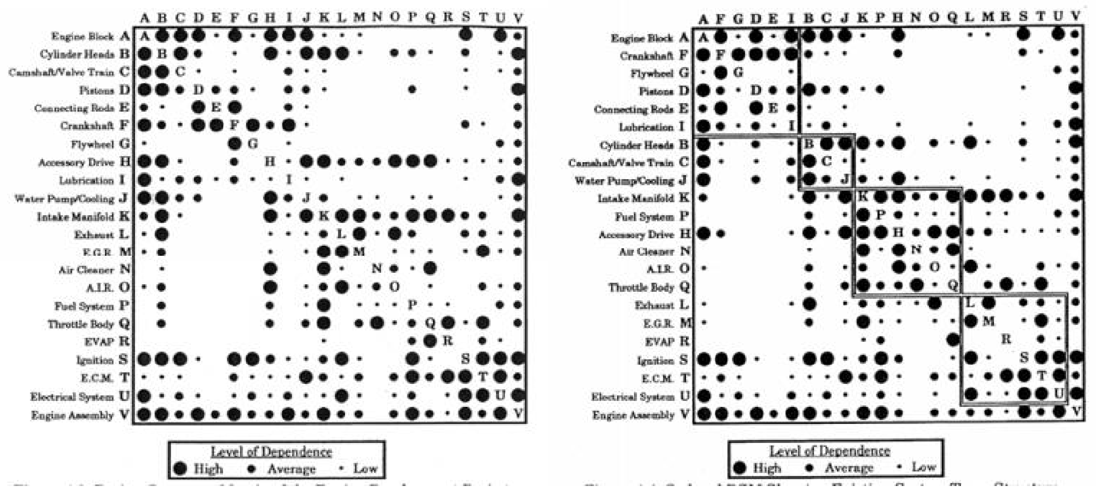


Figure 11. *Design structure matrix of Engine Development Project (McCord & Eppinger 1993)*

Lehtonen (2015, p. 52) states that the clusters in a DSM could also represent modules. The clusters internal dependency is great and independent of the environment. This changes the use of the DSM because in case of modules consisting of parts of the system the matrix cannot be directional. The product structure of modular architecture can be considered static so the it is not important which elements need the other. This changes the matrix into symmetrical form and enables the use of different mathematical methods. According to Lehtonen (2007, p.54) the most used method for arranging the elements is the bandwidth algorithm. Bandwidth algorithm optimizes the order of the elements so that the relations are as close to the diagonal line as possible. This forms a "band" of relations. The algorithm's main task is to reorder the rows and columns so that all marks are as close to the diagonal line as possible. Algorithms can also form tight clusters. There are many algorithms available. (Hölttä & Salonen 2003).

A clustering algorithm can be applied to DSM so that the interactions within clusters are maximized and between the clusters minimized. The formed clusters are possible module candidates. (Hölttä & Salonen 2003) DSM is powerful tool for analyzing method for clustering and sequencing problems (Gong et al. 2017). The use of DSM in analysing product architecture suggests more effective module and subsystem boundaries, highlights critical interfaces and identifies outsourcing opportunities. (Eppinger & Salminen 2001)

According to Pakkanen et al. (2016) DSM is often seen in publications about modularization. DSM is suggested for modelling system architecture and defining modular product architectures (Browning 2001, Helmer et al. 2010).

The DSM can be used to analyze architecture of many different things such as products, organizations and processes. The differences between different DSMs are the used elements. DSM is a tool that captures the dependencies between the elements so different elements are needed for examining different things. Also, the level of granularity of the DSM is dependent on the chosen elements. DSMs can be categorized into static architecture DSMs, temporal flow DSMs and multi-domain DSMs. The Static DSMs divide into product architecture DSMs and organization architecture DSMs. The product architecture DSMs elements can be subsystems, components or functions. The organization architecture DSM focuses on departments, teams and individuals. DSM can also model temporal flow such as process architecture. Process architecture focuses on subprocesses, activities and parameters. Multi-domain DSM collects everything at the same matrix. (Eppinger et al. 2012, pp. 11-12) The different types of DSM models are presented in figure 12.

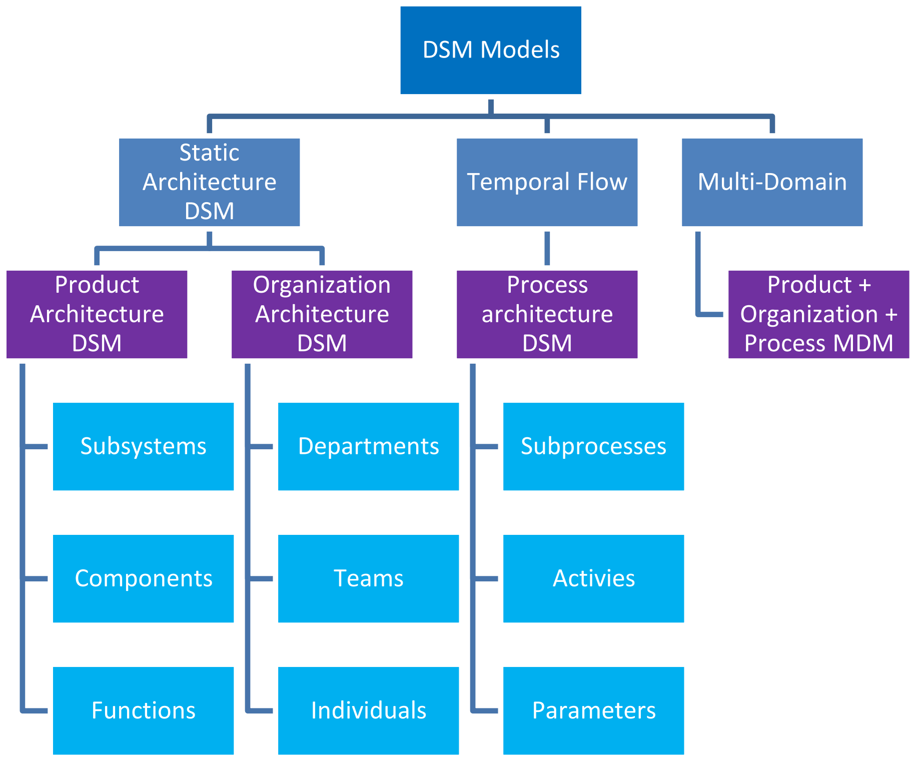


Figure 12. *Different types of Design Structure Matrices (adapted from Eppinger et al. 2012, p. 11)*

3.6.3 Modular Function Deployment

Other method for modularization is the Modular function deployment (MFD) approach. MFD is more customer oriented because it focuses on the strategic business objectives. Product data and information is gathered together into a collection of matrices known as the Product Management Map (PMM). PMM consists of four matrices. The matrices are Quality Function Deployment (QFD), Design Property Matrix (DPM), Module Indication Matrix (MIM) and Interface matrix. (Simpson et al. 2014, p. 95) PMM is presented in figure 13.

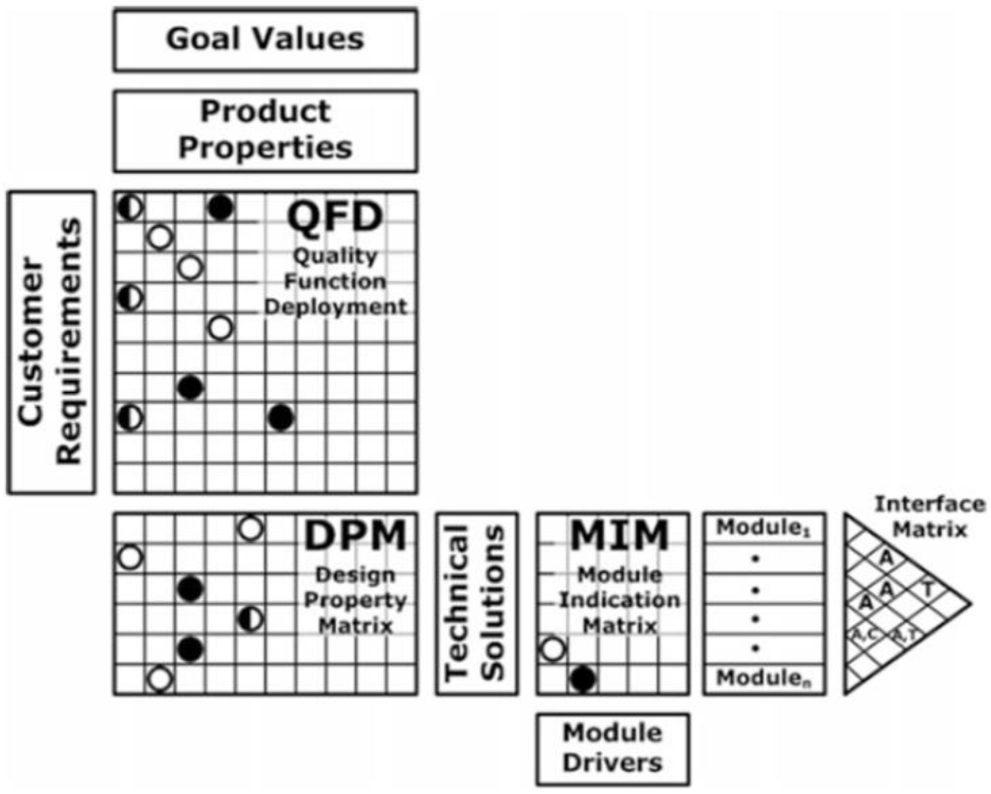
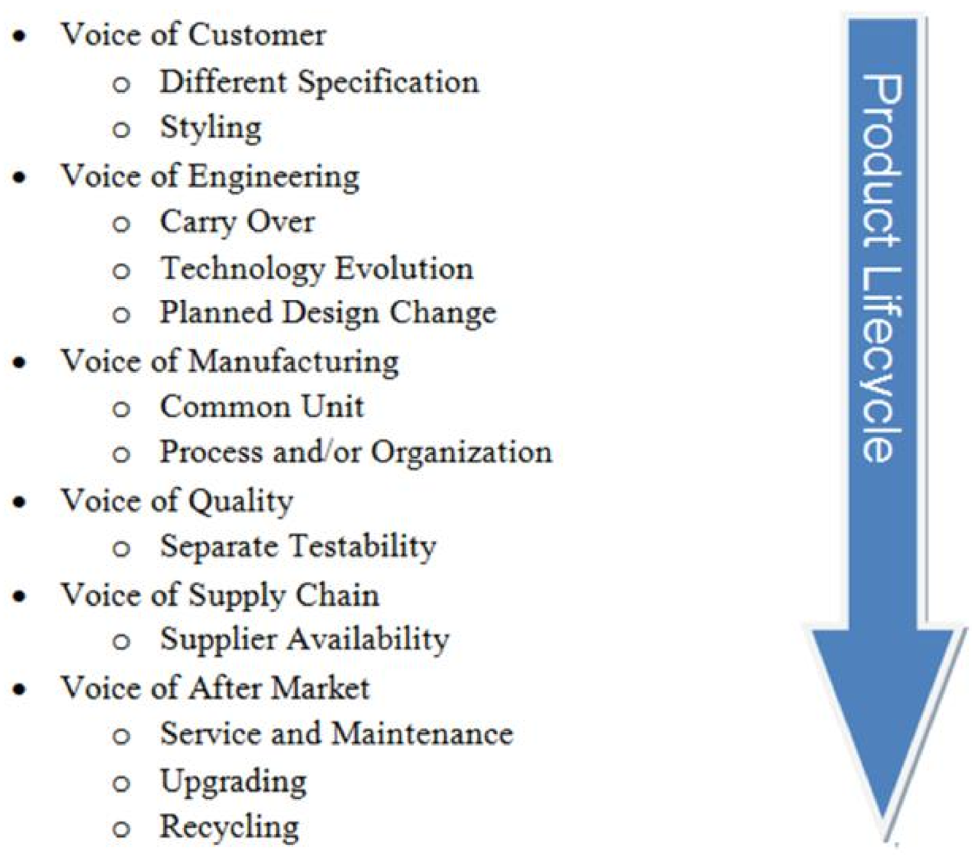


Figure 13. *Product management map (PMM) (Simpson et al. 2014, p. 95)*

According to Erixon (1998) MFD is composed of five basic steps. The steps are:

1. Clarify the customer specifications
2. Select technical solutions
3. Generate concepts
4. Evaluate concepts
5. Improve modules

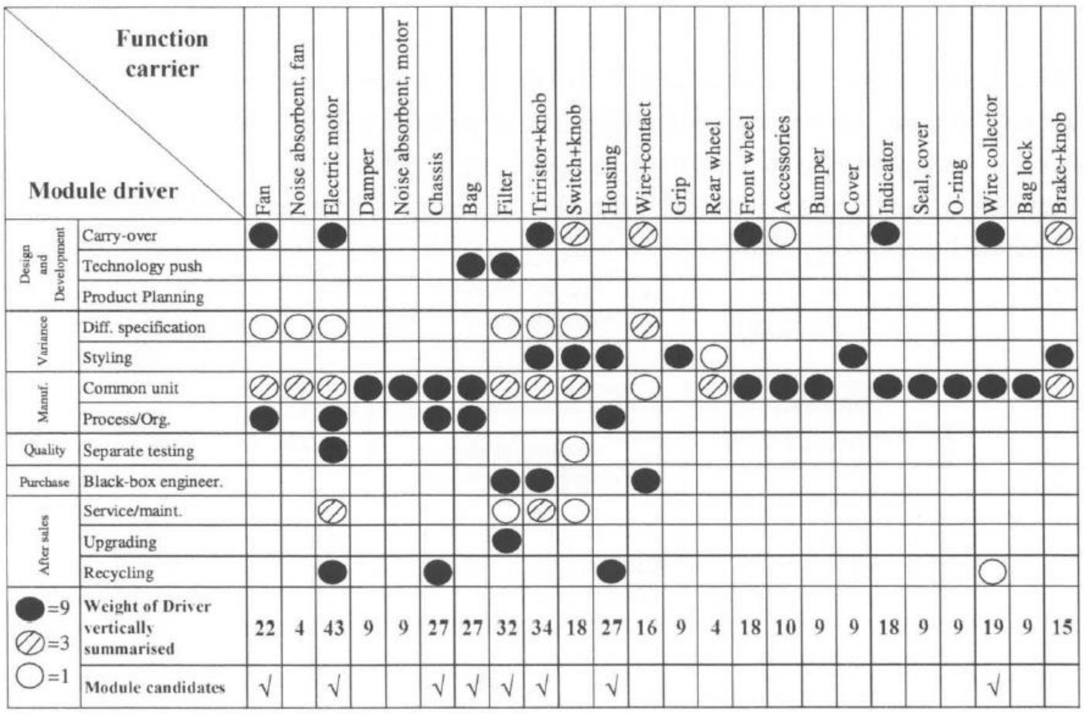
In the first step the customer requirements are mapped against product properties. The tool used in this is the Quality Function Deployment matrix (QFD). The second step is to establish the functional requirements of the product and then decompose the functions into technical solutions. These technical solutions are modeled using the Design Property Matrix (DPM). The third step involves using the module drivers presented in figure 14. The technical solutions are given values according to the drivers. (Simpson et al. 2014 p. 94-95)



- Voice of Customer
  - Different Specification
  - Styling
- Voice of Engineering
  - Carry Over
  - Technology Evolution
  - Planned Design Change
- Voice of Manufacturing
  - Common Unit
  - Process and/or Organization
- Voice of Quality
  - Separate Testability
- Voice of Supply Chain
  - Supplier Availability
- Voice of After Market
  - Service and Maintenance
  - Upgrading
  - Recycling

Figure 14. *List of module drivers (Siivonen 2015, p. 17)*

The Module Indication Matrix (MIM) is presented in figure 15. In MIM the horizontal rows show the parts and in the vertical column are the drivers of modularity. The parts are evaluated according to module drivers and are then given weights. The weight that can be given 1, 3 or 9. The weights are given according to the designer’s knowledge. The designer must decide what is the most important driver for the part and what is not. The module candidates are then suggested according to the weights.



Function carrier / Module driver		Fan	Noise absorbent, fan	Electric motor	Damper	Noise absorbent, motor	Chassis	Bag	Filter	Tiristor+knob	Switch+knob	Housing	Wire+contact	Grip	Rear wheel	Front wheel	Accessories	Bumper	Cover	Indicator	Seal, cover	O-ring	Wire collector	Bag lock	Brake+knob
Design and Development	Carry-over	●		●						●	▨		▨			●	○			●			●		▨
	Technology push							●	●																
	Product Planning																								
Variance	Diff. specification	○	○	○					○	○	○		▨												
	Styling									●	●	●		●	○				●						●
Manuf.	Common unit	▨	▨	▨	●	●	●	●	▨	▨	▨		○		▨	●	●	●		●	●	●	●	●	▨
	Process/Org.	●		●			●	●				●													
Quality	Separate testing			●							○														
Purchase	Black-box engineer.								●	●			●												
After sales	Service/maint.			▨					○	▨	○														
	Upgrading								●																
	Recycling			●			●					●											○		
●=9 ▨=3 ○=1	Weight of Driver vertically summarised	22	4	43	9	9	27	27	32	34	18	27	16	9	4	18	10	9	9	18	9	9	19	9	15
	Module candidates	√		√			√	√	√	√		√											√		

Figure 15. *Module Indication Matrix (MIM) (Erixon 1998, p. 108)*

In the fourth step the module concepts are evaluated by considering how the modules will be physically joined together using standardized module interfaces. The final fifth step improves the module concept with DFX (Design for X) concepts. (Simpson et al. 2014 p. 96)

3.6.4 Function Structure Heuristic method

Function structure heuristic method is a functional decomposition block diagram of all the product's functions and material, energy and information flows between them. The goal is to separate modules from a single product's by using heuristics. (Höttä & Salonen 2003) In literature different heuristics are suggested for separating modules.

- Finding dominant flow, branching flows or conversion-transmission function pairs (Stone et al. 2000)
- Finding similar and repetitive functions within a single product, common functions across products and unique functions that are found in one product within the product family (Zamirowski & Otto 1999)
- Separating causally linked function pairs as modules (McAdams et al. 1999)

The use of this tool starts with a function structure and considering the possible alternative modules that can be defined by group functions according to the heuristics. The modules are then chosen according to the designer’s expertise. (Hölttä & Salonen 2003)

3.7 Ownership of interfaces

Defining ownership of modules and interfaces means defining who has the responsibility for developing documenting and maintaining them. If this is not defined clearly, it might lead to confusion about who has the responsibility. (Harlou 2006 p. 88)

The ownership of modules and interfaces is not widely researched topic in the literature. Although some suggestions are made for ownership of interfaces. Harlou (2006 p. 88) presents three approaches that can used in defining the ownership.

- The owner of the architecture owns the interfaces – First approach is to appoint the ownership and the responsibility of the interfaces to the owner of the architecture. The architecture owner is responsible of architecture development, so it is natural to have the ownership of interfaces also.
- The owner of the standard design owns the interfaces – A second approach is to let the owner of the standard design have responsibility of the interfaces. Each interface is appointed an owner. This will support standardization.
- The third approach is to appoint one owner to each interface. This ownership is independent of the owners of the standard designs and the architectures. The owners of interfaces will develop them in agreement with architecture and standard design.

3.8 Configuration management

Configuration management is a term that is used to describe a process to control items. It can be applied to controlling software and physical products. According to Buckley (1992, pp. 3-4) configuration management is a discipline applying technical and administrative direction and surveillance to:

- Configuration identification
- Configuration change control
- Configuration status accounting
- Configuration audits

Configuration management consists of these four activity areas. All activity areas share metadata for items placed under configuration management. Metadata is a database concept that means data about the data stored in the database. Configuration management process can be viewed as cyclic. The configuration item continuously goes through a loop of revision. (Hass 2003, p. 4)

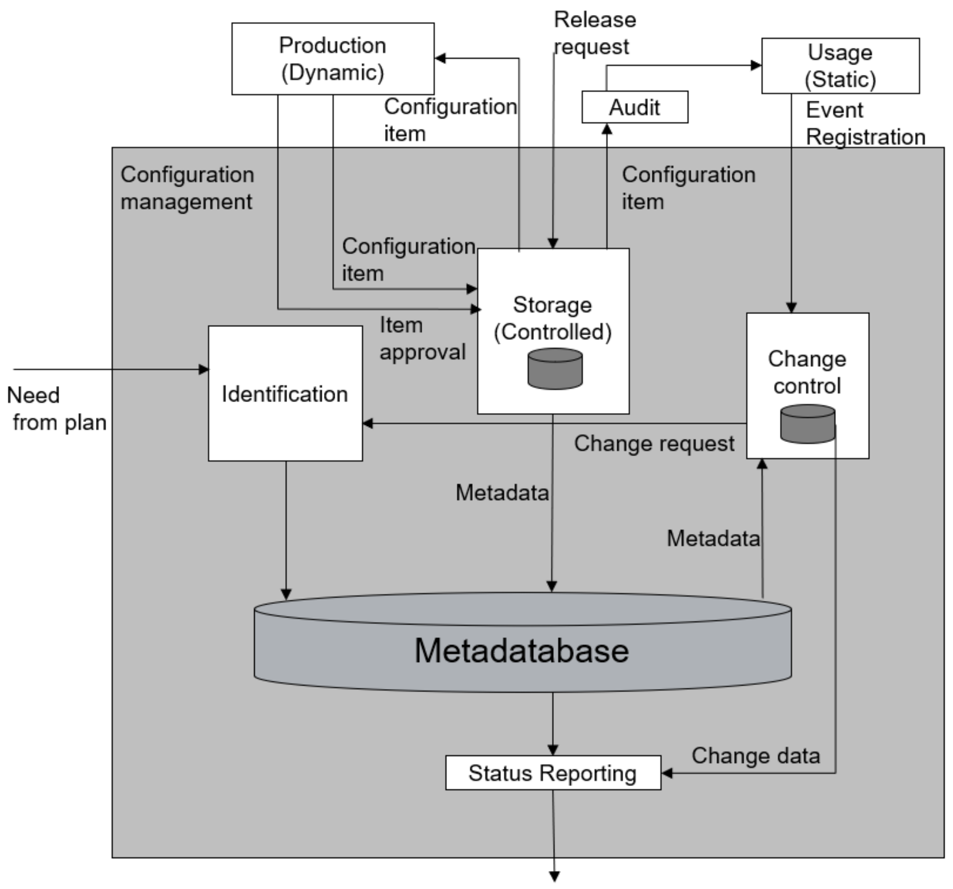


Figure 16. *Configuration management activities (Hass 2003, p. 4)*

When item is put under configuration management it must be identified. Identification is used to determine the metadata that describes the item and to uniquely identify it. After identification the item is subjected to change control. This means that every change made to the item must be traceable. Every change needs to be reported. Change control gets the input for changes from the usage library. The item under configuration management also needs a controlled storage. Items need to have a place where they are kept. The storage is connected to the production library where all changes are made. Status reporting makes data manageable. (Hass 2003, pp. 4-27) Figure 16 shows the configuration management process and the activities involved.

Configuration management practices can be applied to controlling configuration. According to Buckley (1994, pp. 67-68) for establishing effective configuration control three questions need to be answered. The questions are:

1. What needs to be controlled?
2. Who is the configuration control authority?
3. How are the products to be controlled?

The first step in configuration management the identification answers to the first question. If something needs to be controlled, it must always be identified first. Buckley (1994, p. 68) argues that the control authority could be the manager responsible for the product or configuration control board. The controlling of products could be implemented using change documentation such as engineering change proposals (ECP) and engineering change notifications (ECN). Every change needs to leave a trail.

4. THE RESEARCH METHOD AND PROCESS

This chapter presents the research method, research process and used tools. The Design Structure Matrix was chosen as a tool for the modularization of design knowledge.

4.1 Nature of the Research

One goal of this research was to find methods for partitioning of modules. The method for finding the tool was literature review. Modularity and modularization methods have been widely researched. However, modularization as a method for finding modules for design knowledge management has not been addressed much in literature. Ownership of design knowledge is commonly acknowledged method for knowledge management in industrial companies. Still it seems to be very vaguely described in scientific research.

Case study research was chosen for research approach of this study. Case study research method focuses on a single case and uses information obtained using different methods. Case study research method won't restrict the methods used for collecting information and analyzing it so it will be good fit for this kind of research. (Saaranen-Kauppinen & Puusniekka 2006)

The goal of this study was to find the best partitioning of modules so that the modules and their interfaces can be governed efficiently. The research was conducted by generating multiple different Design Structure Matrices. The information that was put into matrices was collected from the experts involved in the design of the components.

4.2 Design Structure Matrix

The tool chosen for finding the design modules in this thesis is the Design Structure Matrix. According to the literature review presented above, DSM is a feasible tool for defining modules and interfaces and visualizing them.

The benefits of modularity that this research is aiming for is the governance of design knowledge and not so much the modularity of the actual product. The drivers for this kind of modularity are the controllability and manageability of modules. This kind of approach to modularity also supports weakening the barriers between organization's units. The other presented modularization methods focus more on the actual product and things like manufacturing and logistics.

A valve topworks assembly is a integral product where the design of other component affects other components. Because of these interdependencies the modules can't be isolated completely. This being the case it is very important to understand what the interdependencies between the modules are. DSM is a good tool for visualizing the design rules that affect multiple components.

For the modules to support the use of design automation the modules must be controllable. The value of the modules will be evaluated according to the ease of governance. Standardizing the interfaces between the modules is crucial for this kind of modularization. If standardization of interfaces is not made, the control of the modules and interfaces will be hard.

The interfaces between the components are crucial because if they are not properly defined the components won't connect with each other. Studying the mechanical dependencies between the components at the right level of granularity visualizes the loose and tight couplings of the components in the system.

Eppinger et al. (2012) present a five-step approach to architectural modelling using DSM. The steps are:

1. Decomposing, in which the system is broken down into its constituent elements.
2. Identifying, in which the relationships among the systems elements are documented.
3. Analysing, in which the elements and relationships are rearranged to understand structural patterns and their implications for system behaviour.
4. Displaying, in which a useful representation of the DSM is created. The DSM need to highlight features of importance.
5. Improving, in which the system is improved through actions taken as a result of the DSM analysis.

These steps will be followed in creating the DSM for this research.

4.3 PDCA-cycle

Deming's PDCA-cycle (Plan Do Check Act) is a method for continuous improvement. It consists of four phases that are planning, doing, checking the results and finally acting on the results. The PDCA-cycle is used in this work to find out the most value creating solution for Metso's problem. In this study the PDCA-cycle was used in making the DSM better.

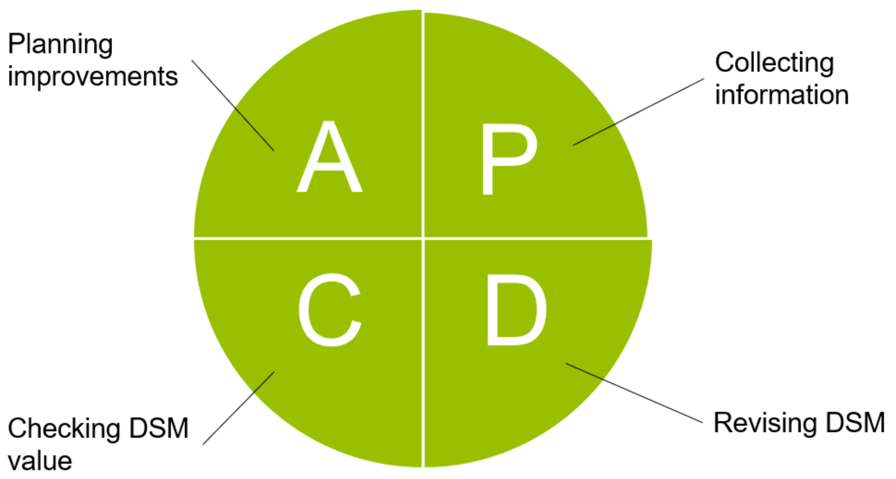


Figure 17. *Process for finding value creating DSM*

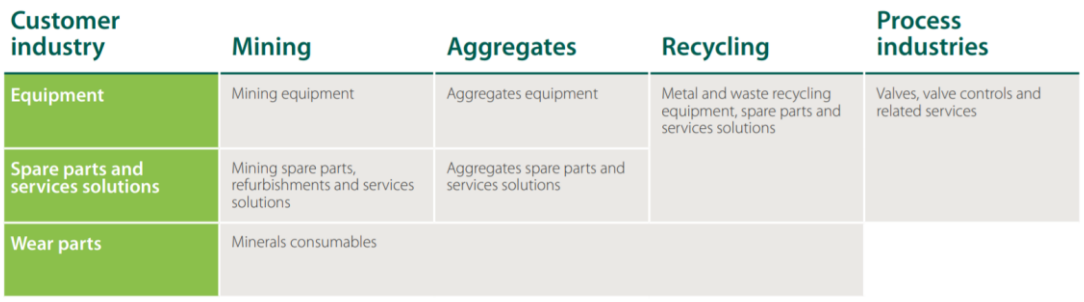
Process presented in figure 17 is suitable for refining the collected data. The weekly iterations allowed the data collected to be valuable for this this research. In the planning phase, more data was collected for the DSM. The DSM was revised in doing phase. In the checking phase the value that the DSM would generate was evaluated. In the acting phase the improvements were planned for the DSM. This iterative cycle continued until the DSM was good and accurate enough.

5. CASE: METSO

This section introduces Metso as a company and presents the products of Metso. The components of the valve topworks assembly will be presented. Additionally, Metso's design automation will be briefly presented. The need for this research will be presented in the last part of this section.

5.1 Target organization

Metso is a Finnish process industry company with products ranging from mining, aggregates and recycling to industrial valves and pumps. Metso has over 13000 employees in more than 50 countries. Metso is listed on the Nasdaq Helsinki and has sales of about 3,2 billion euros in 2018. (Metso 2019) Metso consists of seven business areas: Mining Equipment, Aggregates Equipment, Minerals Services, Minerals Consumables, Recycling, Valves and Pumps. These areas are externally reported under two segments: Minerals and Flow Control. (Metso 2018) Metso's product offerings for different industries are presented in figure 18.



Customer industry	Mining	Aggregates	Recycling	Process industries
Equipment	Mining equipment	Aggregates equipment	Metal and waste recycling equipment, spare parts and services solutions	Valves, valve controls and related services
Spare parts and services solutions	Mining spare parts, refurbishments and services solutions	Aggregates spare parts and services solutions		
Wear parts	Minerals consumables			

Figure 18. *Metso's Product offering (Metso Annual Report: Business Overview 2018, p. 19)*

Metso Minerals' product offering consists of equipment and services for mining, aggregates and recycling. Metso Flow Control offers valves, valve controls and related services for process industries. (Metso 2018 p. 19)

Metso Flow Control produces industrial valve solutions for oil, gas, process and pulp and paper industries. From now on in this thesis Metso Flow Control will be called MFC. MFC's sales were 720 million euros in 2018. (Metso 2018)

This thesis focuses on products of MFC. Typical product of MFC consists usually of valve, actuator, positioner, instrumentation and mounting parts.

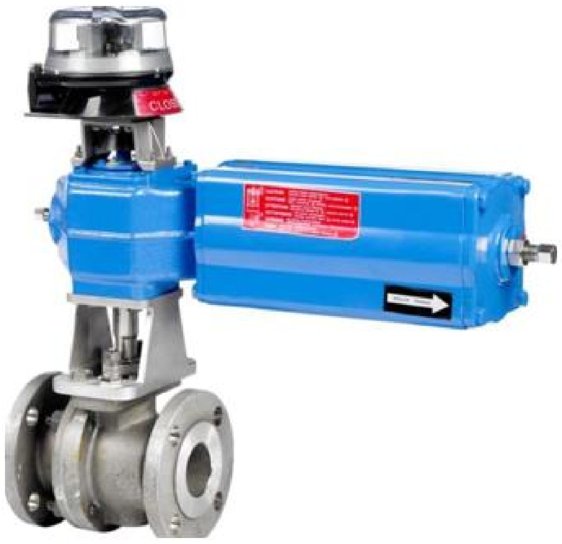


Figure 19. *Valve topworks assembly (Metso 2019)*

Figure 19 present a valve topworks assembly with valve, actuator, limit switch and mounting parts. The main products of MFC are the valve topworks assemblies but MFC also sells valves, actuators and positioners separately.

5.2 Components of the valve topworks assembly

Valve

Valve is a device that regulates, directs or controls the flow of fluid by opening, closing, or partially obstructing various passageways. MFC manufactures many different types of valves like ball valves, butterfly valves and segment valves.

Actuator

Actuator is a mechanism for opening and closing a valve. Actuators can be power-operated or manually operated. Actuators can produce linear and rotational movement. This thesis focuses on quarter-turn pneumatic actuators.

Positioner and Limit switch

Positioner is a device used to increase or decrease the air load pressure driving the actuator. Positioner also measures the position of the valve shaft. Limit Switch is a visual sign which indicates whether the valve is open or closed.

Instrumentation

Instrumentation consists of pneumatic components, piping and fixing plate that connects the components to actuator. Different pneumatic components are needed for actuator to work as planned.

Mounting parts

Mounting parts consist of two brackets, their screws and key coupling used in fitting the valve shaft with actuator. Brackets are used in connecting valve to actuator and positioner and limit switch to actuator.

5.3 Metso’s design automation and the need for this thesis

The design automation is used to automatically produce 3D-models, assemblies and drawings using Siemens PLM Software’s Rulestream ETO design automation software and SolidWorks CAD-software. The design knowledge of valve design has been collected into design rules. The design rules can be pressure vessel design standards, internal calculation methods etc. The design rules control the geometric parameters of the CAD-models.

The design rules are divided into general rules and product specific rules. General rules can be for example standards that affect all product families. Product specific rules affect only specific product family. The next step in the project is to expand the design automation to valve topworks assemblies. The design knowledge related to the other components in the valve topworks assembly needs to be collected into design rules.

At the start of this thesis MFC has not defined clear ownership of the valve topworks assembly’s components design knowledge. The design knowledge of components will be collected into rules just like it has been collected in valves. MFC has a need for definition of modules that the valve topworks assembly comprises of. The design knowledge needs to be separated into modules because the governance of the rules needs to be simple. The governance can be executed by determining the owners of the modules.

MFC also doesn’t have a method for automating CAD-assembly of valve topworks assembly. Assemblies have been made manually but still the process for making assemblies has not been standardized. This leads to problems with variation in assemblies.

The main products of MFC are the valve topworks assemblies. MFC also sells valves, actuators and positioners separately. Because of this the design of the components has been made separately. The results of this thesis could also potentially weaken the organizational boundaries between the different components design teams.

6. USING DESIGN STRUCTURE MATRIX

Theory of Domains provides a very interesting view of the system. It allows the designer to step away from the existing components and to examine the system as a whole. The valve topworks assembly is innately assembly based modular product.

The research started by using DSM at the part domain to understand the relations between the components. It is important to point out that this thesis views the valve topworks assembly as the system and the parts of the system are the components such as valve and actuator. These components can be considered as systems of their own but that is not relevant in the scope of this research.

The dependencies between elements are described using marks. This kind of DSM is static, so the matrix becomes symmetric. The marks are added to both sides of the diagonal line for ease of clustering. Clustering is made by separating the clusters of marks with boxes.

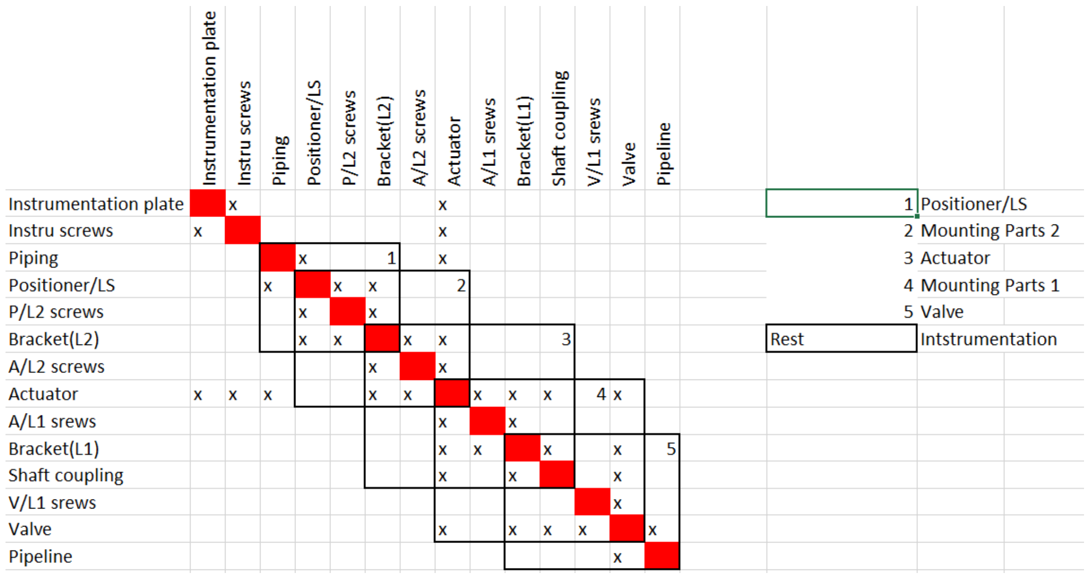
The first DSM produced examined the dependencies of the components at the part domain. DSM of part domain is presented in figure 20. This kind of DSM focuses only on the mechanical dependencies. The only deduction that can be made from this kind of DSM is that valve topworks assembly is a very integral product. No modules can be derived from this kind of presentation.

	Valve	Actuator	Pos/LS	Instrumentation	Standard parts
Valve	🟥	x			x
Actuator	x	🟥		x	x
Pos/LS			🟥	x	x
Instrumentation		x	x	🟥	
Standard parts	x	x	x		🟥

Figure 20. *First DSM*

The second DSM produced also examined the dependencies of the components at the part domain. The standard parts entity that comprises of brackets and key coupling was split into these parts. The second DSM is presented in figure 21. By examining the system more closely and adding piping and screws to the DSM, a much better presentation of the system and its interdependencies can be found. The modules suggested by this DSM are presented below.

1. Positioner/LS
2. Linkage 2
3. Actuator
4. Linkage 1
5. Valve
6. Instrumentation



	Instrumentation plate	Instru screws	Piping	Positioner/LS	P/L2 screws	Bracket(L2)	A/L2 screws	Actuator	A/L1 srews	Bracket(L1)	Shaft coupling	V/L1 srews	Valve	Pipeline
Instrumentation plate	🟥	x						x						
Instru screws	x	🟥						x						
Piping			🟥	x		1		x						
Positioner/LS			x	🟥	x	x		2						
P/L2 screws				x	🟥	x								
Bracket(L2)				x	x	🟥	x	x			3			
A/L2 screws						x	🟥	x						
Actuator	x	x	x			x	x	🟥	x	x	x	4	x	
A/L1 srews								x	🟥	x				
Bracket(L1)								x	x	🟥	x		x	5
Shaft coupling								x		x	🟥		x	
V/L1 srews												🟥	x	
Valve								x		x	x	x	🟥	x
Pipeline													x	🟥

1	Positioner/LS
2	Mounting Parts 2
3	Actuator
4	Mounting Parts 1
5	Valve
Rest	Intstrumentation

Figure 21. *DSM of part domain with connecting parts*

The second DSM is good enough for modules to be clustered from it. The boxes in figure 21 present the clusters of dependencies. These can also be thought as modules. The addition of screws and its effect on the DSM highlights that for standardizations sake the design data of screws must be managed somehow. However, this kind DSM does not give much insight to what needs to be controlled in the interfaces.

It is important to remember that the goal of this research is to find out the best way to partition this system into modules that are easy to govern. Studying the part domain does not help in determining what needs to be governed. Studying the organ domain of the system does not provide any better view of the interdependencies of system. By studying the function domain can the significant independencies be found.

To make a function DSM of the system, the function structure of system must first be clarified. Defining the function structure starts with finding the main functions of the system. The found main functions are:

- Connecting valve to pipeline
- Valve external tightness
- Valve internal tightness
- Transferring torque to trim
- Transferring torque to valve
- Connecting valve shaft to actuator
- Connecting actuator to valve
- Actuator internal tightness
- Connecting controller to actuator
- Connecting actuator to controller
- Operating actuator
- Controlling position
- Connecting plate to actuator
- Transferring air

These functions were found by discussing with MFC's experts and by using the authors own expertise. These functions also mirror the customer requirements for the valve topworks assembly. The components themselves have many functions that contribute to the main function of the component as a system. The main functions collected for the valve topworks assembly's function structure are relevant for the functionality of the valve topworks assembly. It is possible to go deeper into the functions of each component but that is not relevant within the scope of this research.

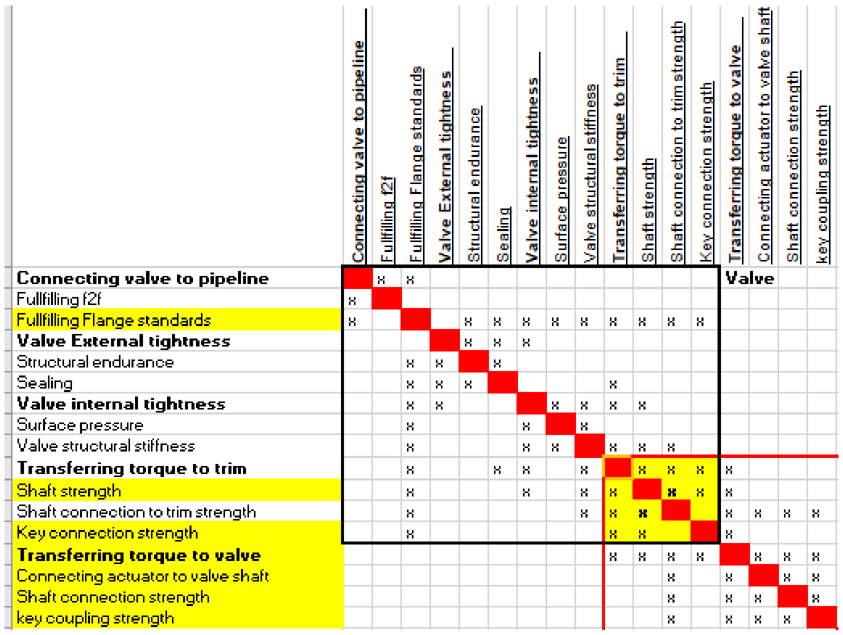
The main functions presented above compose of subfunctions. Examining these subfunctions and their dependencies is the level of granularity needed for understanding the system. The main functions division into subfunctions is presented in table 1. The subfunctions provide some understanding on the type of design knowledge that needs to be controlled. For example, the first function *Connecting valve to pipeline* can be split into subfunctions *Fulfilling face to face* and *Fulfilling flange standards.* The valve cannot be connected to the pipeline without these two functionalities. This is also relevant in the case of valve topworks assembly. If the valve topworks assembly can't be connected to the pipeline, then it is useless.

Table 1. *Functions and subfunctions*

Connecting valve to pipeline	Fullfilling facetoface
	Fullfilling Flange standards
Valve External tightness	Structural endurance
	Sealing
Valve internal tightness	Surface pressure
	Valve structural stiffness
Transferring torque to trim	Shaft strength
	Shaft connection to trim strength
	Key connection strength
Transferring torque to valve	Connecting actuator to valve shaft
	Shaft connection strength
	key coupling strength
Connecting valve shaft to actuator	Fitting valve shaft with actuator
	Enduring torque
Connecting valve to actuator	Connection structural strength
	Enabling packing
Connecting actuator to valve	enabling heat protection for actua-tor
	Enduring actuator weight
	Enduring actuator torque
Actuator Internal tightness	generating force
	Generating torque
Connecting controller to actuator	positioner vibration resistance
	enabling use of outsourced actuator
Connecting Actuator to controller	
Connecting controller to actuator shaft	
Operating actuator	Operating speed control
	Controlling actuator air pressure
	suplying air
Controlling position	Information transfer
	Reacting to information
	Position measurement
Connecting plate to actuator	Pneumatic component fixing
Transferring air	

The DSM that represents the dependencies between the systems functions and sub-functions is presented in appendix A. The function DSM allows for more clear presenta-tion of the system. From the function DSM multiple different suggestions for modules were made by clustering the dependencies. DSM is very versatile tool that allows the user to derive different solutions from it. The value of the solutions is up to the user to understand. The clustering presented in appendix A was deemed to support knowledge management because it mirrors the existing product structure and the organization's structure. Thus, governance of such modules would be easier to establish. This clustering divides the valve topworks assembly into modules that are valve, linkage 1, actuator, linkage 2, positioner and limit switch and instrumentation.

The valve module's functions are presented in figure 22. Valve is the most independent component in the system. It connects to actuator through linkage 1. The most important point is that the function *Fulfilling flange standards* is also related to the design of torque transmission and thus to the design of linkage 1. This creates a clear interface between the modules.



	Connecting valve to pipeline	Fulfilling f2f	Fulfilling Flange standards	Valve External tightness	Structural endurance	Sealing	Valve internal tightness	Surface pressure	Valve structural stiffness	Transferring torque to trim	Shaft strength	Shaft connection to trim strength	Key connection strength	Transferring torque to valve	Connecting actuator to valve shaft	Shaft connection strength	key coupling strength
Connecting valve to pipeline	🟥	x	x											**Valve**			
Fullfilling f2f	x	🟥															
Fullfilling Flange standards	x		🟥		x	x	x	x	x	x	x	x	x				
Valve External tightness				🟥	x	x	x										
Structural endurance			x	x	🟥	x											
Sealing			x	x	x	🟥				x							
Valve internal tightness			x	x			🟥	x	x	x	x						
Surface pressure			x				x	🟥	x								
Valve structural stiffness			x				x	x	🟥	x	x	x					
Transferring torque to trim			x			x	x		x	🟥	x	x	x	x			
Shaft strength			x				x		x	x	🟥	x	x	x			
Shaft connection to trim strength			x						x	x	x	🟥		x	x	x	x
Key connection strength			x							x	x		🟥	x			
Transferring torque to valve										x	x	x	x	🟥	x	x	x
Connecting actuator to valve shaft												x		x	🟥	x	x
Shaft connection strength												x		x	x	🟥	x
key coupling strength												x		x	x	x	🟥

Figure 22. *Valve module*

Linkage 1 could also be part of the valve module. MFC has tried to standardize the mounting parts so controlling those at their own module is reasonable. Separating the mounting parts from other components and appointing an owner for them supports standardization. The functions clustered for the linkage 1 module are presented in figure 23. The main functions that contribute to connecting valve to actuator and actuator to valve have different subfunctions. Therefore, it is reasonable to deduct that because linkage 1 affects valve and actuator in different ways it should be controlled as its own module. The interfaces between linkage 1, valve and actuator discovered examining the overlapping areas of the clusters.

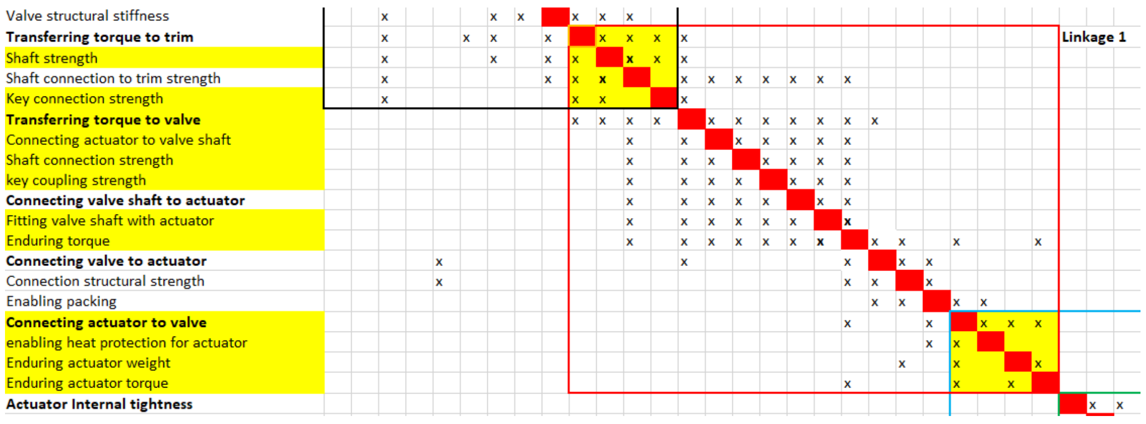

Valve structural stiffness
Transferring torque to trim
Shaft strength
Shaft connection to trim strength
Key connection strength
Transferring torque to valve
Connecting actuator to valve shaft
Shaft connection strength
key coupling strength
Connecting valve shaft to actuator
Fitting valve shaft with actuator
Enduring torque
Connecting valve to actuator
Connection structural strength
Enabling packing
Connecting actuator to valve
enabling heat protection for actuator
Enduring actuator weight
Enduring actuator torque
Actuator Internal tightness
Linkage 1

Figure 23. *Linkage 1 module*

Actuator module is presented in figure 24. Actuator is the component that is most connected to other components. Actuator module has interfaces with all other modules. The actuator connects to valve via linkage 1 and the key functions for the connection are *connecting valve to actuator* and *connecting actuator to valve*. These are interfaces that affect all three modules. These functions affect the design of mounting faces.

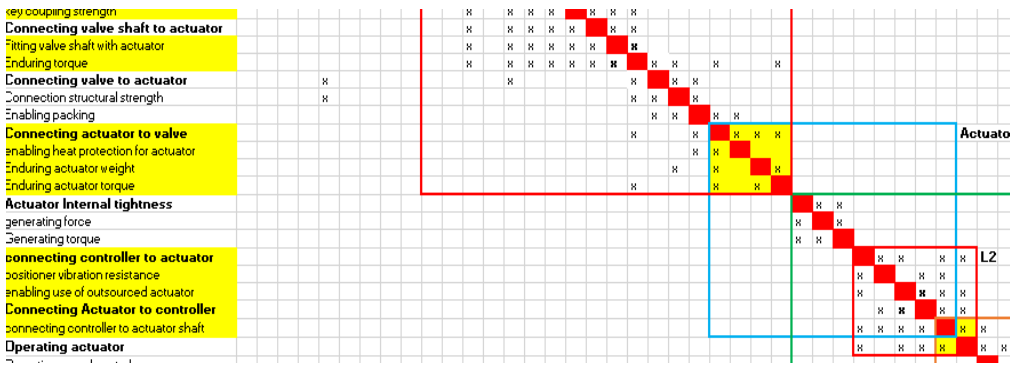

Connecting valve shaft to actuator
Fitting valve shaft with actuator
Enduring torque
Connecting valve to actuator
Connection structural strength
Enabling packing
Connecting actuator to valve
enabling heat protection for actuator
Enduring actuator weight
Enduring actuator torque
Actuator Internal tightness
generating force
Generating torque
connecting controller to actuator
positioner vibration resistance
enabling use of outsourced actuator
Connecting Actuator to controller
connecting controller to actuator shaft
Operating actuator
Actuato
L2

Figure 24. *Actuator module*

Figure 25 presents the modules instrumentation, linkage 2 and positioner/limit switch. Linkage 2 connects actuator to positioner and limit switch. Separating linkage 2 into its own module offers the same benefits of standardization as in the case of linkage 1. Instrumentation connects to actuator and positioner through *transferring air* function. Instrumentation module cluster is not as clear as the other modules. In figure 23 the instrumentation cluster encompasses actuator, linkage 2 and positioner/limit switch clusters.

In reality the design of instrumentation does not affect the design of linkage 2. One characteristic of the DSM is that in cases where the elements in DSM cannot be reorganized to cluster the dependencies into separate boxes, the user can decide to do what was done with the instrumentation cluster. The important part is that the user can understand what has been done. The interfaces that are critical for instrumentation module are related to *transferring air* function.

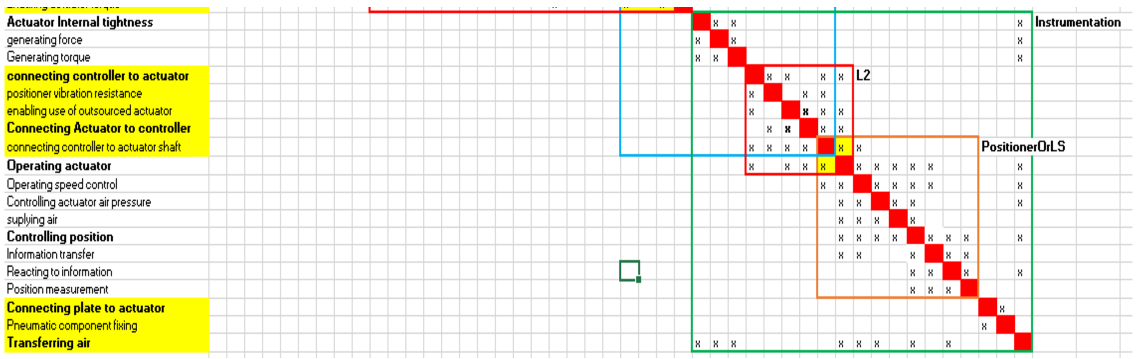

Actuator Internal tightness
generating force
Generating torque
connecting controller to actuator
positioner vibration resistance
enabling use of outsourced actuator
Connecting Actuator to controller
connecting controller to actuator shaft
Operating actuator
Operating speed control
Controlling actuator air pressure
suplying air
Controlling position
Information transfer
Reacting to information
Position measurement
Connecting plate to actuator
Pneumatic component fixing
Transferring air
Instrumentation
L2
PositionerOrLS

Figure 25. *Instrumentation, Linkage 2 and Positioner/Limit switch modules*

The main contribution of the use of DSM in describing the valve topworks assembly was the understanding that the components are not truly independent. Therefore, the governance of interfaces must be emphasized. The design module interfaces could be discovered using the function DSM. The function DSM provided a valuable solution for module partitioning.

7. DEFINED DESIGN MODULES, INTERFACES AND GOVERNANCE

The defined design modules and interfaces are presented in this chapter. The governance model for defined modules and interfaces will be introduced. The assembly skeleton created according to the modules will also be presented.

7.1 Design modules

The defined modules that control the design of components are valve, actuator, positioner and limit switch, instrumentation, linkage 1 and linkage 2. The modules containing the mounting parts linkage 1 and linkage 2 could also be combined to a standard parts module. The author's opinion is that the standard parts module is too vague for knowledge management purposes. The design of mounting parts presented in chapter 5.3 needs to be divided into separate modules. Linkage 1 module consists of the shaft coupling and the bracket between valve and actuator. Linkage 2 module consist only of the bracket between positioner/limit switch and actuator.

The modules defined by using DSM are presented in figure 26. The addition to the modules that are involved in design of the components, a new interfaces-module was established. Collecting the interfaces into their own module turns away from the typical modular structure. The motivation for doing this comes from the goal of this research, the knowledge management. Collecting all interface design rules into same place allows for stricter control of these important rules. The interfaces module consists of rules that affect more than one module. These rules are crucial for the functionality of valve topworks assembly.

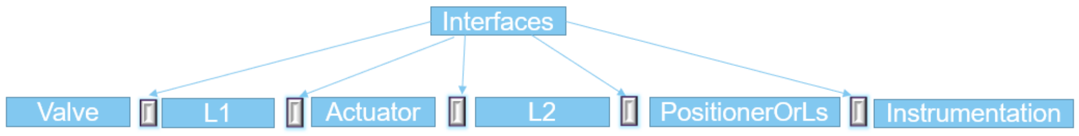


Figure 26. *Defined design modules*

In figure 26 the blocks between the modules represent the interfaces. The arrows coming from the interfaces-module represent transfer of design rules. It is noteworthy to mention that there is no design rule transfer between the component modules. The module that controls the design knowledge at these interfaces is at higher level than the component modules. Collecting the design knowledge into the interfaces-module removes the interactions between the component modules. This makes the component modules independent from each other. This forms only loose couplings between the components and thus enables modularity of design knowledge.

These modules can be developed independently. This is a clear benefit of modularity that can be achieved by dividing the design knowledge into these modules. The modules valve, actuator, positioner/limit switch and actuator mirror the organizational structure of MFC. This helps in establishing the ownership for these modules. MFC already has persons who have been appointed the responsibility of their design. But still clear ownership has not been defined.

The mounting part modules linkage 1 and linkage 2 do not yet have an appointed owner. Author's opinion is that recognizing mounting parts as important design modules and appointing owners for them is important. This leads to standardization and thus enables the modularity of valve topworks assembly.

7.2 Design interfaces

The interfaces appear whenever the changes in design of another module affects the other module. By making the DSM of the function structure of the system the design rules that affect multiple modules could be identified. Example of the design interfaces are the design of mounting faces between valve and actuator and actuator and positioner and limit switch. These design interfaces need to be governed separately from modules because errors in them lead to the system not functioning anymore.

7.3 Ownership of modules

For the governance of the modules it is crucial to define the ownership of the modules. Ownership defines the who has the responsibility of the maintenance of the design knowledge.

The component design modules should be governed by the owner of the architecture of each component. The owner is required to have a thorough understanding about the component architecture. The component architecture owner also needs to have high enough status in the organization. The architecture owner will be responsible for all general rules related to the component's architecture. All the changes related to architecture owner's component module go through him or her.

The ownership of the interfaces-module should be given to a team composed of the owners of component architecture. This will be called Architecture Interface Control Board (AICB). The design changes concerning multiple components need to be discussed with all affected components architecture owners. Figure 27 presents the ownership of the modules.

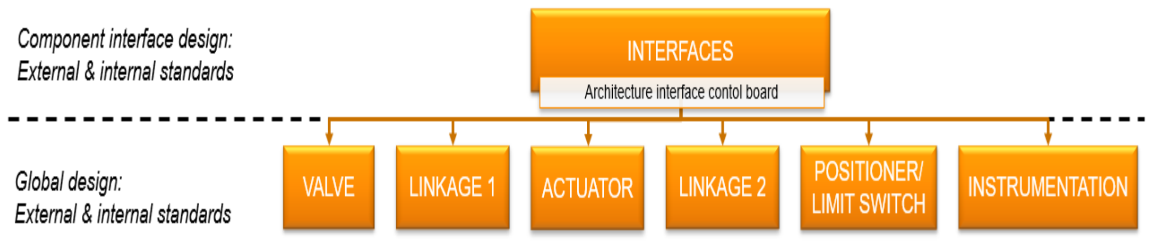

Component interface design:
External & internal standards
INTERFACES
Architecture interface contol board
Global design:
External & internal standards
VALVE
LINKAGE 1
ACTUATOR
LINKAGE 2
POSITIONER/ LIMIT SWITCH
INSTRUMENTATION

Figure 27. *Ownership of the modules*

The understanding of the component architecture ensures that the impact of changes in rules is understood by the architecture owner. This allows the owner to make decisions inside the module. The architecture owner also needs to have good communication channels to other modules owners. The communication between different architecture owners might contribute to concurrent engineering. This could manifest itself in better sharing of design knowledge between design teams.

Finding a person with these kinds of qualities in MFC’s organization will not be easy. The owner needs to have advanced knowledge of design of the component. Otherwise the owner won’t understand how the general design rules affect the different product families.

The mounting part modules linkage 1 and linkage 2 are not the same as the other modules. These modules are separated from the other components mainly because of standardization. The architecture of these components is so simple that the design knowledge ownership of linkage 1 can be given to owner of for example valve architecture or to the actuator architecture owner. The ownership of linkage 2 can be given to positioner/limit switch architecture owner or actuator architecture owner.

7.4 Assembly skeleton

Assembly skeleton was created using the defined design modules and interfaces. Assembly skeleton will be used in automating the assembly of valve topworks assembly in SolidWorks and in documenting the crucial mechanical interfaces.

The modules and interfaces were used in making an assembly skeleton of the valve topworks assembly. This assembly skeleton also documents the mechanical interfaces that are needed for assembling the valve topworks assembly. The assembly skeleton that was generated according to the results of this thesis is presented in figure 28.

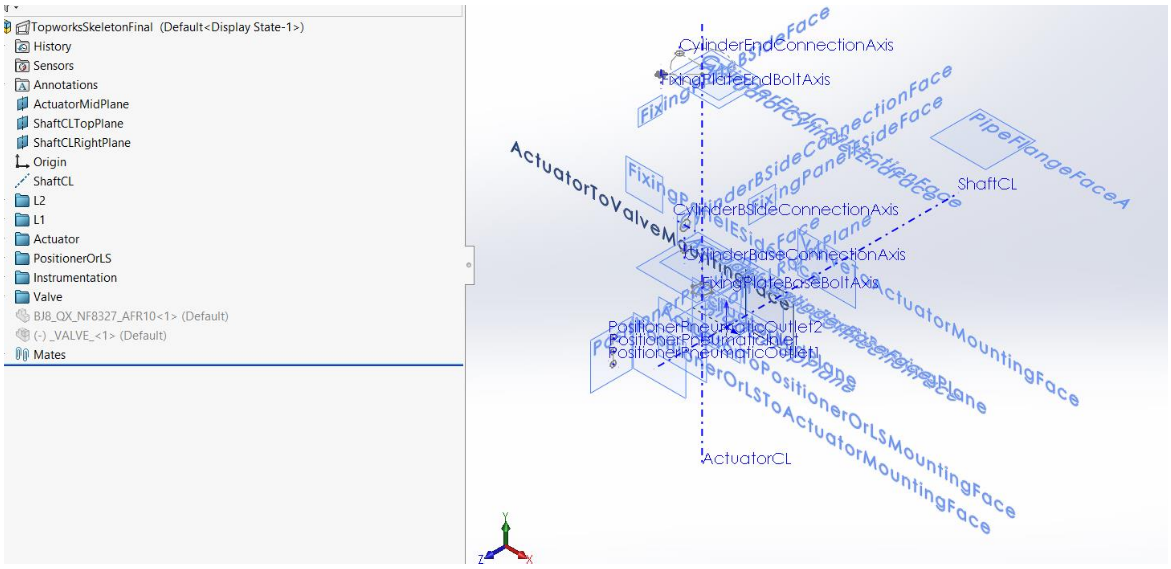


Figure 28. *Assembly skeleton*

The skeleton mirrors the defined modules and their interfaces. The Skeleton constructs from reference geometry such as planes and axises. The defined modules that are in the skeleton contain the reference geometry needed for the component to find its place in the assembly. Because the reference geometry is divided into modules, it is possible to change the reference geometry of one component without affecting to other components.

The reference geometry for components also needs to be documented into separate modelling instructions. The design automation won’t work if the models of components don’t contain the reference geometry defined in the assembly.

The naming conventions of these interfaces has been made in cooperation with components designers. The names are meant to be comprehensible for everyone working with valve topworks assembly. The naming is crucial for the design automation because the software recognizes everything by name. The naming is not presented in this thesis.

Figuring out names for interfaces and needed reference geometry is an asset to the organization. The mutual naming conventions help in understanding each other between teams. in global company where there are multiple teams working with same components the comprehensible naming can save time and effort.

8. RESULTS AND SUGGESTIONS

This chapter presents the results of the research in summary and will answer to the research questions presented in chapter 2.1. Suggestions for future research are made at the end of this chapter.

8.1 Results

This thesis defined the design modules of the valve topworks assembly that will be used in governing design knowledge that will be collected into design rules. DSM was the tool chosen for partitioning of the modules. DSM was chosen according to the literature review. The granularity needed for decomposing the system into modules that are valuable in the scope of design knowledge governance was found by making many different matrices. The function domain proved to be accurate enough for highlighting the important interfaces between the modules. The dependencies relevant for the research were found by cooperating with MFC's experts. The defined modules are valve, actuator, positioner/limit switch, instrumentation, linkage 1 and linkage 2. The interfaces that are important in the scope of design knowledge governance are the design rules that affect multiple modules. Each module will have an appointed owner titled component architecture owner. Different modules owners form Architecture Interface Control Board. The board is responsible for the interfaces. Assembly skeleton was created for automating CAD-assembly of valve topworks assembly. The assembly skeleton is divided into defined modules and it contains all relevant mechanical interfaces.

8.2 Research questions

According to the literature review and the function DSM it is possible to answer the research questions presented in chapter 2.1.

1. *How to define design modules that support the knowledge governance?*

DSM is a suitable tool for defining modules that support knowledge governance. The function domain offers accurate enough view of the system. The modules that support knowledge governance need to be independent from each other. Defining the modules requires extensive knowledge of the system.

2. *What are the interfaces between the modules and how they can be discovered?*

The design knowledge interfaces are rules that affect multiple modules. Interfaces can be discovered by clustering the function DSM into modules. The interfaces appear when clusters overlap or when element has dependencies with multiple clusters.

3. *How should the design modules and interfaces be governed?*

The governance of the design modules and interfaces is important. Every design module needs to have an appointed owner that is responsible for the module. The interfaces-module's governance must be emphasized. The interfaces-module needs to be governed by the Architecture Interface Control Board.

8.3 Discussion and suggestions

The goals of this research were achieved. The defined modules will support the design knowledge governance. The final DSM that was made visualizes the dependencies between the components. The research confirmed that the components are not truly independent. The defined component modules will be governed by the architecture owners and the interfaces module will be governed by the AICB. The assembly skeleton mirrors the design knowledge modules, so the use of the skeleton helps in implementing the modular structure of the design knowledge.

This research was a case study. The limited time for the research restricted the implementation of the governance model and design modules. Gathering the information for the DSM could have been more systematical. The nature of the tool DSM gives the user freedom to interpret the matrix in many ways, so the modules defined is only one possible solution. The defined modules were deemed to support design knowledge governance so other alternatives are not needed.

Collecting the functions for the function structure of the system was not easy because designers involved in the design of their own components often were thinking only about their own component. Many designers seem to have limited knowledge of other components in valve topworks assembly. The PDCA cycle worked very well in this kind of iterative research. Examining and enhancing the DSM weekly allowed it to be accurate enough.

The DSM is a tool that is only as good and accurate as the data that is inputted. Collecting the information into the DSM was conducted in collaboration with the designers of MFC so if the dependencies are not accurate then the organization doesn't have good understanding of the architecture of its product. The ownership of design modules is not a widely researched topic. Much of the decisions regarding the ownership were made by the author. Determining the owners will make the responsibilities concerning the general design rules clearer.

The author suggests that MFC could benefit from creating the function structure of every component and dividing the components into design modules. This could provide advanced insight of the component's architecture. Dividing the design into modules could also speed up the development of components.

The ownership model suggested in this thesis needs to implemented well. This requires that the reasons leading to this change need to be explained clearly to all affected people. MFC could benefit from closer cooperation with different components design teams. Also, the clear responsibilities on governing the design knowledge removes the confusion about who has the responsibility.

The use of modularity in design knowledge management is not a widely researched topic in literature. The benefits for dividing the design knowledge of somewhat integral product into manageable modules include enabling concurrent engineering, ease of knowledge management and easier maintenance of design knowledge. It will also support standardization. The way of dividing the modules to mirror the organizational structure makes it easier to implement the ownership of the modules.

Design automation is a trending subject in the manufacturing industry. Rule-based design of components brings new challenges to design knowledge management. The errors in rules multiply themselves into all products so it is crucial to focus on the governance of the rules. This research presents a solution for dividing this kind of design knowledge into manageable modules and the process is applicable for similar cases.

9. SUMMARY

The goal of this thesis was to define design knowledge modules and interfaces for design knowledge management. The governance of the modules was also determined. The ownership of the modules and interfaces was defined to support the governance of modules and interfaces. The method for modularization of design knowledge was chosen according to literature review. The Design Structure Matrix was suitable tool for defining the modules. The interfaces were found at the intersection points of the clusters. This thesis created a base for design knowledge management in MFC.

The mechanical interfaces can be documented into assembly skeleton. The design modules will be controlled by component architecture owners. The design interfaces that affect several modules need to be governed strictly. The architecture interface control board will manage this crucial design knowledge.

The presented results were based on the literature review and authors own suggestions. In literature modularity has been widely researched but the processes for this kind of modularization and for creating governance model for design knowledge needs further research.

In the timeframe of this thesis it was not possible to implement the design modules and the governance model. It would be interesting to see how the implemented modules will work in practice. The future of design automation in MFC is tied to how well the governance of the design knowledge is implemented. Author thinks that companies in manufacturing industries that adopt design automation will need to determine things that were determined for MFC in this thesis in the future.

REFERENCES

Akao, Y. (1990). Quality Function Deployment, QFD - Integrating Customer Requirements into Product Design. Productivity Press, Portland, ON, USA.

Andreasen, M. M. (1980) Syntesemetoder på systemgrundlag – bidrag tile n konstruktionsteori, Doctoral Dissertation, Lunds Universitet.

Andreasen, M. M. (2011). 45 years with design methodology. *Journal of Engineering Design, 22*(5), pp. 293-332.

Baldwin, C. Y., & Clark, K. B. (1997). Managing in an age of modularity. Harvard Business Review Sep/Oct97, Vol. 75 Issue 5, pp. 84-93.

Borjesson, F. & Hölttä-Otto, K. (2013). A module generation algorithm for product architecture based on component interactions and strategic drivers. Research in Engineering Design, Jan 2014, Vol. 25(1), pp. 31-51.

Borowski, K. (1961). Das Baukastensystem in der Technik, 1st ed. Springer-Verlag Berlin Heidelberg, 106 p.

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, Vol. 48(3), pp. 292-306.

Buckley, F. J. (1992). Implementing configuration management: Hardware, Software, and firmware, IEEE Computer Society Press, Los Alamitos, USA. 249 p.

Eppinger, S. D & Salminen V., (2001), Patterns of product development interactions, ICED 01 Glasgow

Eppinger, S. D., & Browning, T. R. (2012). Design structure matrix methods and applications. MIT Press, Cambridge, USA. 334 p.

Erixon, G., (1998). Modular Function Deployment - A Method for Product Modularisation, The Royal Institute of Technology, 188 p.

Fujimoto, T. (2007). Competing to Be Really, Really Good: The Behind-The-Scenes Drama of Capability-Building Competition in the Automobile Industry, 1st English ed. ed. International House of Japan, Tokyo, Japan, 156 p.

Garud, R. & Kumaraswamy, A., (1993). Changing competitive dynamics in network industries: An exploration of sun microsystems' open systems strategy. Strategic Management Journal, Vol. 14(5), pp. 351-369.

Gong, X., Liu, Y., & Jiao, R. J. (2017). Variety-driven assembly system layout design by design structure matrix clustering analysis. Procedia CIRP, Vol. 63, pp. 295-300.

Harlou, U. 2006, “Developing product families based on architecture – Contribution to a theory of product families”, Dissertation, Technical University of Denmark, Lyngby, Denmark, 173 p.

Hass, A. M. J. (2003). Configuration management principles and practice (1. print. ed.). Boston, MA: Addison-Wesley. 370 p.

Helmer, R., Yassine, A. & Meier, C., (2010), Systematic module and interface definition using component design structure matrix, Journal of Engineering Design, Vol. 21(6), pp. 647-675.

Höltta, K., Salonen M., (2003). Comparing three modularity methods. I P ASME Design Engineering Technical Conferences C, & IL S. (2003). Publication II.

Lehtonen, T. (2007). Designing modular product architecture in the new product development Tampere University of Technology, Tampere, 220 p.

Malmqvist, J., (2002). A classification on matrix-based methods for product modeling, proceedings of Design 2002-conference, Dubrovnik, 2002. pp. 203-210.

McAdams, D. A., Stone, R. B., & Wood, K. L. (1999). Functional interdependence and product similarity based on customer needs. Research in Engineering Design, Vol. 11(1), pp. 1-19.

McCord K. R. & Eppinger, S. D., (1993). Managing the integration problem in concurrent engineering. Working Paper no.3594, MIT. Sloan School of Management, Cambridge, MA

Metso, (2019). Available: https://www.metso.com/company/metso-in-brief/

Metso Annual Report, (2018). Metso Oyj, 48 p. Available: https://www.metso.com/siteassets/annual-report/2018/metso_business_overview_2018.pdf

Miller, T. & Elgård, P. (1998). Defining Modules, Modularity and Modularization: Evolution of the Concept in a Historical Perspective, Design for Integration in Manufacturing, 1998, Fuglsoe, 19 p.

Pakkanen, J. (2015). Brownfield Process: A Method for the Rationalisation of Existing Product Variety towards a Modular Product Family, Tampere University of Technology. Publication; Vol. 1299, Tampere University of Technology.

Pakkanen, J., Juuti, T. & Lehtonen, T. (2016). Brownfield Process: A method for modular product family development aiming for product configuration, Design Studies, Vol. 45 pp. 210-241.

Parslov, J.F. & Mortensen, N.H. (2015). Interface definitions in literature: A reality check, Concurrent Engineering, Vol. 23(3), pp. 183-198

Perera, H. S. C., Nagarur, N., & Tabucanon, M. T. (1999). Component part standardization: A way to reduce the life-cycle costs of products.International Journal of Production Economics, Vol. 60(1), pp. 109-116.

Pimmler, T. U. & Eppinger S. D., (1994). Integration Analysis of Product Decompositions, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA

Pine, J. (1993). Mass Customization: The New Frontier in Business Competition, Harvard Business School Press, Boston, Massachusetts, 333 p.

Saaranen-Kauppinen, A. & Puusniekka, A. Tapaustutkimus, Luku 5.5. KvaliMOTV - Menetelmäopetuksen tietovaranto, Available:https://www.fsd.uta.fi/menetelmaopetus/kvali/L5_5.html.

Sanchez, R. & Mahoney J. T., (1996). Modularity, flexibility, and knowledge management in product and organization design. Strategic Management Journal, Vol. 17(SPEISS), pp. 63-76.

Siivonen, P., (2015). Modularization of an existing product family, Tampere University of Technology, 71 p.

Simpson, T., Jiao, J., Siddique, Z. & Hölttä-Otto, K., (2013). Advances in Product Family and Product Platform Design: Methods and Applications. New York: Springer.

Steward, D. V., (1981). The design structure System: A Method for managing design of Complex systems. IEEE Transactions of Engineering Management, Vol. 28(3), 1981, pp. 71-74.

Stone, R. B., Wood, K. L., & Crawford, R. H. (2000). A heuristic method for identifying modules for product architectures. Design Studies, Vol. 21(1), 5-31.

Sudjianto, A. & Otto, K., (2001). Modularization to Support Multiple Brand Platforms, Proceedings ASME DETC-2001, Paper No DETC2001/DTM-21695, Pittsburgh, PA, USA

Suh, N. P., (1990). Principles of Design, Oxford University Press, Cambridge, UK, 1990.

Ullman, D., (1992) The Mechanical Design Process. New York: McGraw-Hill.

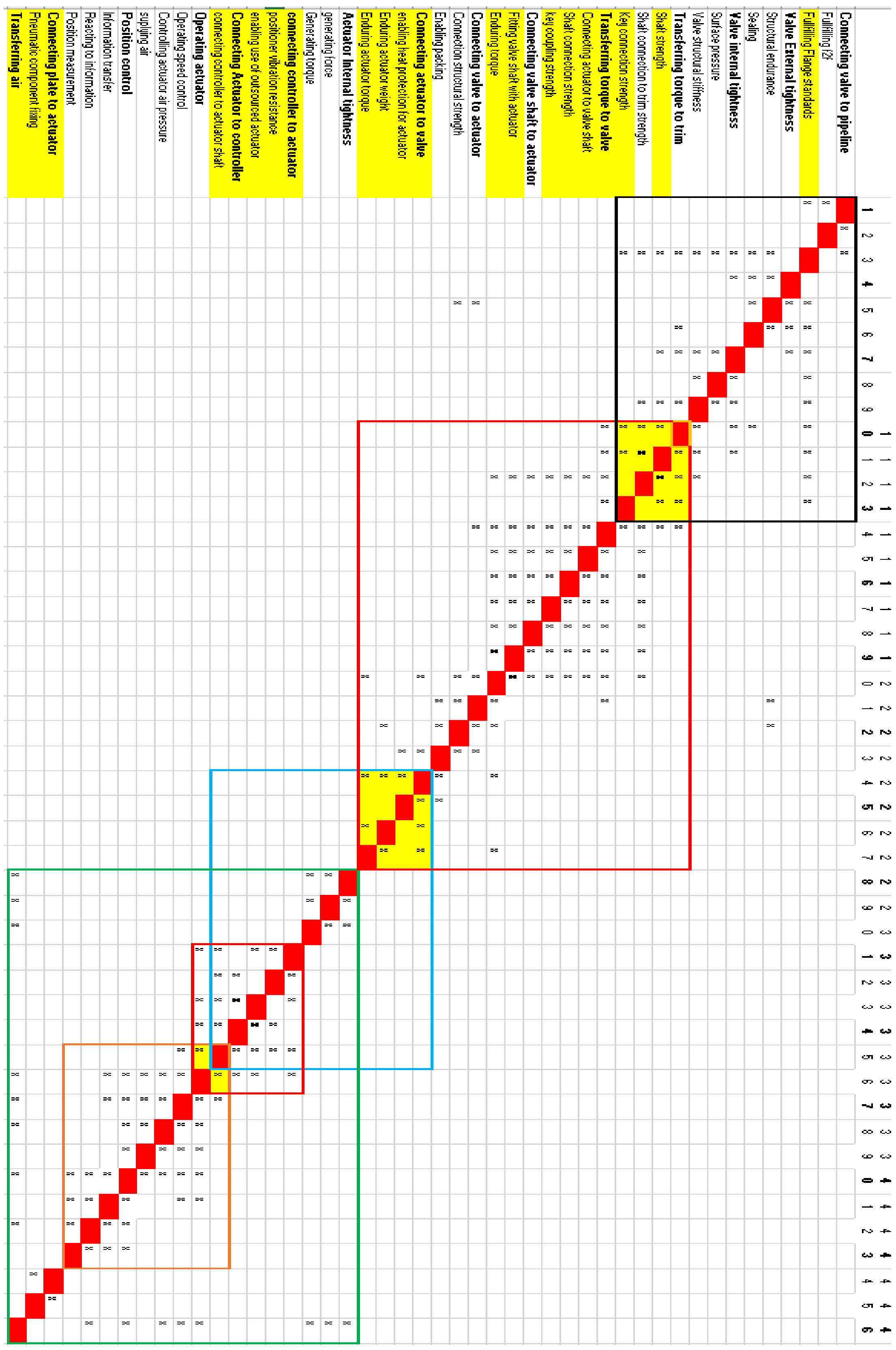
Ulrich, K (1995) The role of product architecture in the manufacturing firm. Research Policy, Vol. 24(3): 419–440.

Ulrich, K. T., & Tung Karen. (1991). Fundamentals of product modularity. Proceedings of the 1991 ASME Winter Annual Meet- Ing Symposium on Issues in Design/Manufacturing Integration, ASME, New York, NY.

Ulrich, K. T. & Eppinger, S. D., (2012). Product design and development, 5th ed. McGraw-Hill/Irwin. New York, NY, 415 p.

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.

APPENDIX A: FUNCTION DSM



Connecting valve to pipeline
Fulfilling f2f
Fulfilling Flange standards
Valve External tightness
Structural endurance
Sealing
Valve internal tightness
Surface pressure
Valve structural stiffness
Transferring torque to trim
Shaft strength
Shaft connection to trim strength
Key connection strength
Transferring torque to valve
Connecting actuator to valve shaft
Shaft connection strength
key coupling strength
Connecting valve shaft to actuator
Fitting valve shaft with actuator
Enduring torque
Connecting valve to actuator
Connection structural strength
Enabling packing
Connecting actuator to valve
enabling heat protection for actuator
Enduring actuator weight
Enduring actuator torque
Actuator Internal tightness
generating force
Generating torque
connecting controller to actuator
positioner vibration resistance
enabling use of outsourced actuator
Connecting Actuator to controller
connecting controller to actuator shaft
Operating actuator
Operating speed control
Controlling actuator air pressure
suplying air
Position control
Information transfer
Reacting to information
Position measurement
Connecting plate to actuator
Pneumatic component fixing
Transferring air