ADAPTING DSM CLUSTERING TO THE MODULARIZATION OF DIVERSE AND UNIVERSAL PRODUCT FAMILIES

A Thesis

by

KYLE BRADLEY BAYLIS

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Daniel A. McAdams
Committee Members,	Michael D. Johnson
	Astrid C. Layton
Head of Department,	Andreas A. Polycarpou

May 2017

Major Subject: Mechanical Engineering

ABSTRACT

Product family design is a popular approach for designing a group of related products to strategically share common features and components. One method for developing product families is by using a combination of shared and unique modules. The success of a modular product family largely depends on the proper selection of modules and module boundaries. While a number of methods exist for the modularization of individual products, many of these methods are not currently suited for use with product families. The objective of this research is to develop a method for extending the use of popular component-based modularization methods to product families. This thesis primarily consists of two distinct manuscripts.

In the first manuscript, a method for extending the use of DSM clustering to the modularization of diverse product families is presented. In this approach, the modular architecture of the product family is optimized while also maximizing commonality between products. A Pareto front is developed of different architectures that produce optimal strategic modularity and maximized commonality in the product family. The proposed method is applied in a case study to the design of a product family of power tools. In this case study, the quality of the modular architecture is evaluated using a DSM (Design Structure Matrix) for each product. Three architectures along the Pareto front are chosen and examined to demonstrate the usefulness of the technique.

The second manuscript presents an approach that incorporates the use of the proposed modularization method in the design of universal product families. The approach utilizes market segmentation techniques and action-function modeling to identify the special design requirements for disabled users. An algorithmic approach is employed to generate modular architecture alternatives for constructing the detailed product family. The approach is demonstrated using a case study over the design of typical and inclusive vehicle driver seats.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a thesis committee consisting of Dr. McAdams (advisor) and Dr. Layton of the Department of Mechanical Engineering and Dr. Johnson of the Department of Manufacturing and Mechanical Engineering Technology.

Funding Sources

This graduate study was supported in part by the National Science Foundation. Its contents are solely the responsibility of the author and do not necessarily represent the official views of the National Science Foundation.

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

In an effort to efficiently meet the demands of diverse user groups, products are often designed as a part of product families. A product family consists of a set of product variants that share common elements or features (Simpson et al. 2006). Product families typically provide a range of financial and organizational benefits at the cost of product distinctiveness and performance (Cameron and Crawley 2014). In product family design, increased commonality is often equated with a variety of benefits. However, there exists a trade-off between product distinctiveness and commonality (Robertson and Ulrich 1998). Proper selection of product family architecture and common components enables high commonality without overly compromising distinctiveness. Figure 1 demonstrates the trade-off between the distinctiveness of products and commonality for three different product family architectures. A primary objective of product family design is to identify an architecture that allows for high commonality in the family with minimal losses to product distinctiveness and performance.

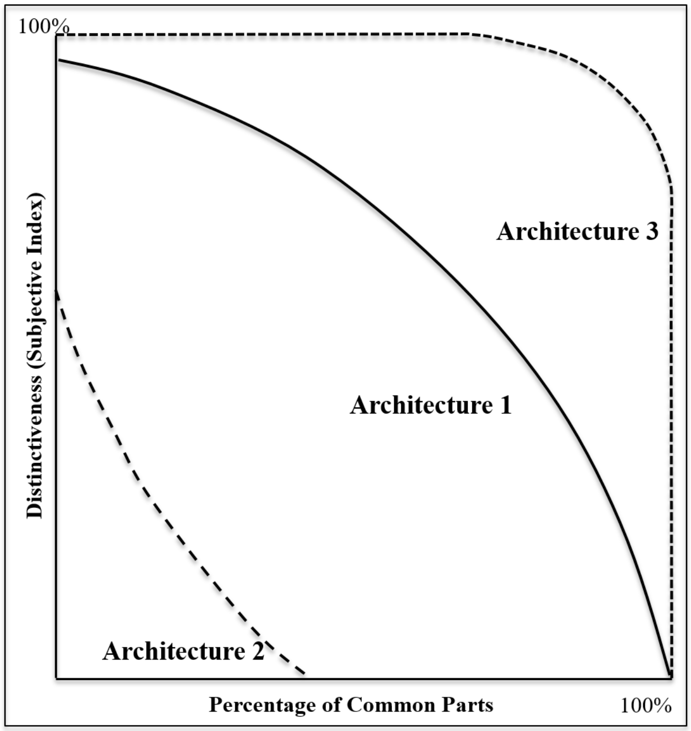


Figure 1: Tradeoff between distinctiveness and commonality (recreated from Robertson and Ulrich 1998)

Most product families can be categorized into two types: parametric product families and modular product families (Simpson et al. 2006). Parametric product families share common design features that are parametrically scaled to modify performance. On the other hand, modular families utilize a combination of shared and unique modules to configure each product variant.

The design of a modular product family requires the selection of modules that will provide maximal commonality while maintaining the configurability and other design benefits associated with modular design. Modules are subassemblies within a product that ideally possess high coupling between the components within and low interaction with other external modules. There exist several prominent methods for modularizing individual products; however, many of these approaches are not readily applicable to the design of product families (Hölttä and Salonen 2003).

Due to its efficacy in providing variety within product line, product family design naturally lends itself to use in universal design. Universal design is a design approach that seeks to create products that are usable and accessible to all persons (Mace 1985). Universal product families leverage common elements to configure products that meet the needs of both typical and disabled users. These product families consist of universal modules (shared by all products), typical modules (providing features for typical persons), and accessible modules (providing necessary features/functionality for disabled users) (McAdams and Moon 2012). Universal product family design enables the design of products that address the needs of all users, without the development challenges of providing an all-inclusive product.

This paper focuses on a proposed method for the design of the modular architecture of a product family. The method was developed in an effort to extend popular modularization methods to the design of modular product families. This method applies modularization at a detailed design level of product design. Architecture alternatives are evaluated as a trade-off between maximum commonality and the quality of the modular architecture. Previously developed commonality indices are employed to assess the commonality provided by potential modular architectures. A DSM clustering algorithm is utilized to develop an index for the quality of the modular architecture on the basis of maximizing internal module connectivity and minimizing external connectivity between modules. Using this approach, a Pareto front of various modular architectures that provide maximum commonality and optimal modular architecture is identified, along which the final design can be selected and refined.

This paper is organized into two stand-alone manuscripts. The first manuscript, in Section 2, introduces the proposed method and provides a case study for the design of a product family of power tools. Section 3 presents a strategy to the design of universal product families. In this manuscript, the proposed modularity method is extended, in conjunction with other universal design methods, to use in designing universal product families. The recommended strategy is demonstrated using a case study of the design of typical and accessible vehicle driver seats. Overarching conclusions from the manuscripts are presented in Section 4.

2 DESIGN OF MODULAR PRODUCT FAMILIES AS A TRADEOFF BETWEEN COMMONALITY AND QUALITY OF MODULAR ARCHITECTURE

2.1 Introduction

In an effort to enable mass customization, designers have increasingly begun designing products around product families. A product family is a group of similar product variants that share one or more common elements. Product families offer a cost-effective solution for providing a variety of products to meet the needs of diverse markets (Simpson et al. 2006).

Ulrich (1995) identified two types of architectures from which products are typically built: modular and integral. Integral product architecture maps the functional elements of a product to a single or small number of physical components. Conversely, modular structures possess one or more modules to accomplish each product function. Modular architectures afford flexibility in altering the design of a product. When functional requirements change, only the module/modules related to those functions would need to be altered or replaced. Consequently, modular product architecture is often utilized in the design of product families. Modular product families consist of a combination of shared and unique modules from which each product variant is constructed.

Simpson et al. (2006) provide evidence that modular architecture can be an effective and cost-efficient method for creating customizable products. Several primary methods exist for generating modules for individual products. Methods for modularizing products can be categorized into functional-based and component-based methods. Several prior methods have been developed for the functional-based module identification for product families, however, there is a lack of methods that extend the component-based modularization methods to use with product families.

In this paper, a method is presented for generating modular architecture alternatives that lie along a Pareto front of maximum commonality and optimal strategic modular architecture. The following sections include background and prior work in the field of modular and modular product family design, an explanation of the proposed method, a case study to provide validation of the method, and, finally, conclusions on the method with ideas for future work.

2.2 Background

2.2.1 Product Family Design

A group of products that shares a product platform to satisfy a variety of market niches is a product family. A product platform is a group of common components that are shared by multiple products or generations of products. Using a product platform of common components/modules, multiple products can be efficiently developed (Meyer and Lehnerd 1997). Research has shown product platform design to be a cost-effective option for providing variety in products, allowing designers to meet the needs of multiple market segments (Simpson et al. 2006). However, platforming itself does not necessarily create an advantage. Careful planning must go into the architecture of the product platform to ensure that it provides a design advantage.

The success of a product platform is largely dependent on the balance of the tradeoffs between commonality and variety. The commonality of a shared platform has both benefits and drawbacks. In general, there are typically economic benefits and performance losses associated with sharing a product platform. The typical goal of product family design is to maximize commonality without sacrificing product distinctiveness and performance. The benefits of commonality can be broken into three categories: revenue benefits, cost savings, and risk benefits (Robertson and Ulrich 1998). The costs of commonality can affect many aspects of the design. Primarily, designs that implement commonality strategies generally require a larger initial investment for a more rigorous design process than individual product designs. However, there are also a variety other costs that may be realized throughout the lifecycle of the design. Cameron and Crawley produced a list of costs and benefits that occur in five phases of the product lifestyle-Strategy, Design, Manufacturing, Testing, and Operations (Cameron and Crawley 2014). A number of different commonality indices exist for quantifying the amount of commonality within a product family. Thevenot and Simpson (2006) produced a comparison of six prevalent commonality indices and provided recommendations for the use of each index.

Modular platform design is often employed in the development of product families. In modular product family design, the product family is designed around a combination of shared and unique modules that are combined to construct each product variant. Over the years, a variety of methods have been developed for architecting a modular platform design. To aid in the selection of methods for applying product platform design, Otto et al. (2013) grouped the design process into twelve generic activities and defined available methodologies for accomplishing each action.

2.2.2 Modular Product Design

Products with modular architecture consist of easily distinguished blocks of components called modules. An ideal modular architecture exhibits high coupling within modules and low coupling between modules. The success of modular designs is largely dependent upon the proper choices of modules, module boundaries, and interfaces.

Ulrich and Tung (1991) discuss the potential costs and benefits of modular product design. Potential benefits arise from design and production economies, customer responsiveness, and the organization and operation of design and production systems. The benefits they reported include component economies of scale, product updating and variety, decreased order lead-time, improved design, production, and testing, and ease of maintenance and repair. The costs of modularity include static production architecture limiting innovation, decreased performance, ease of reverse engineering by competitors, increased unit variable costs, and excessive product similarity.

A number of different competing methods exist for designing an individual product's modular architecture. Some of the predominant methods include using functional heuristics, Design Structure Matrix (DSM) clustering algorithms, and Modular Function Deployment (MFD) heuristics. Function structure heuristics allow for module identification at the functional level, whereas DSM clustering and MFD strategic heuristics are more suited for implementation at the component level.

Stone et al. (2000) introduced a method for using the function structure of a product to identify potential functional modules. This technique is advantageous as it can be applied early in the design at a functional level, before concept development has taken place. The method utilizes three module heuristics: dominant flow, branching flow, and conversion-transmission. Figure 2 presents the three function structure heuristics.

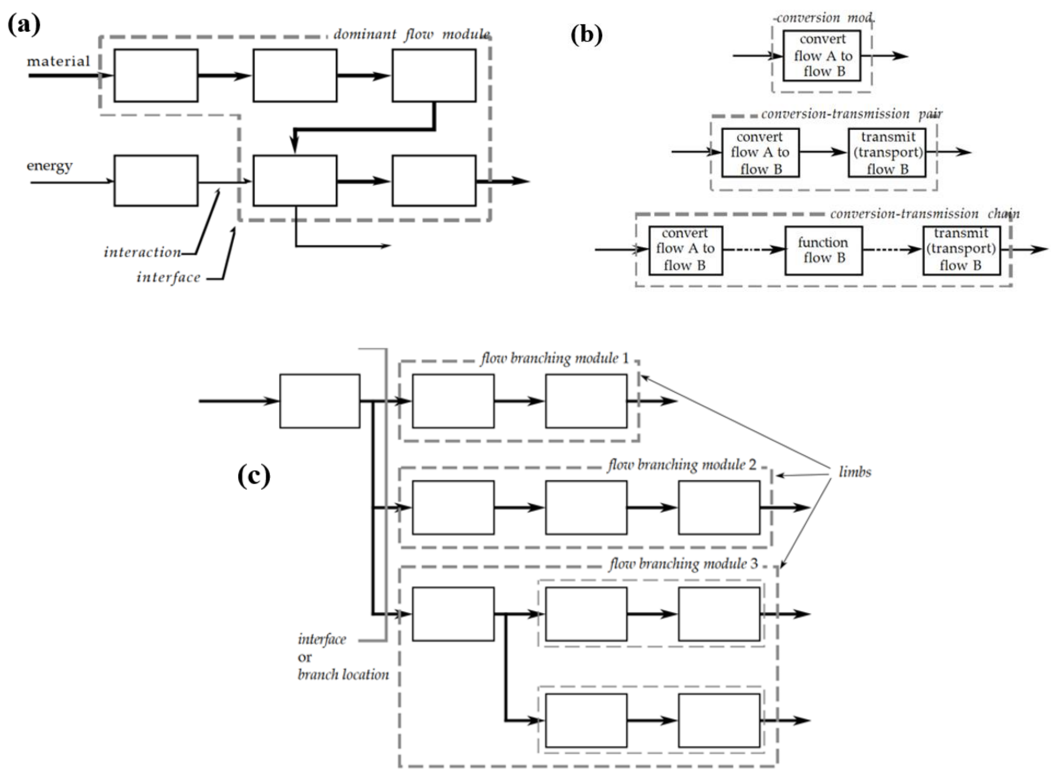


Figure 2: Module heuristics; (a) dominant flow, (b) conversion-transmission, (c) branching flow (Stone et al. 2000)

Module heuristics consistently identify more modules than the number ultimately included in the finalized product. The decision of modules to implement is based on expected costs, customer needs, and other design-specific strategies. Function structure heuristics prioritize grouping components into function-based modules.

A Design Structure Matrix (DSM) is a commonly used tool for defining the module boundaries of a product. DSM clustering was first introduced by Steward (1981) for use in the management of designing complex systems. Since then, DSM has been adapted for grouping and organizing product components (Eppinger and Browning 2012). DSM is most often implemented after the generic system architecture and/or components have been defined. In modular design, a DSM matrix represents the interactions between the components of a product. Using this matrix, components can be clustered into modules.

A variety of clustering algorithms have been developed for DSM clustering (Thebeau 2001; Helmer et al. 2010; Yu et al. 2007). Interactions in the DSM may be specified using either binary or weighted terms, depending on the algorithm used. Eppinger (1997) introduced a method for using DSM to cluster product components into modules using either generic interactions or differing degrees of material, spatial, energy and information interactions. DSM algorithms are designed to group components into modules that contain strong interactions between components within the module and little to no interactions with other modules. Figure 3 presents an example of the DSM method of clustering.

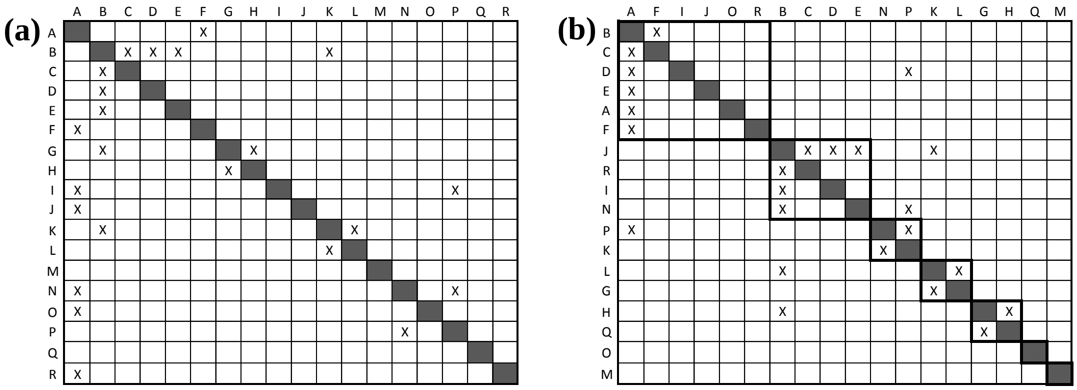


Figure 3: DSM clustering example; (a) unclustered matrix, (b) clustered matrix

Predominant DSM clustering algorithms operate by attempting to minimize a cost function associated with the clustering. This cost function is typically a combination of costs caused by interactions between components that are not in the same cluster, and costs derived from low interactions between components within the same cluster. The algorithm attempts to minimize the clustering cost by randomly selecting a component and assigning it to a new cluster. The clustering ends when the algorithm has processed a set number of attempts without finding a clustering change that will decrease the clustering cost.

Another method for identifying potential modules is Modular Function Deployment (MFD). This approach to modularity was first described by Erixon (1998). Systematic MFD begins with quality function deployment (QFD), used to gather customer requirements and identify related functional requirements. Technical solutions are subsequently identified to fulfill the functional requirements. To organize technical solutions into strategic modules, a Module Indication Matrix (MIM) is constructed to assign "Module Drivers" to each technical solution. Module Drivers consist of 12 heuristics, reported by Östgren (1994), based on the strategic reasons for which designers typically organize product components into modules. Each module may be classified as having one or more Module Driver that defines it. Figure 4 shows an example of a MIM for the modular design of the inner roof of a vehicle.

Module driver \ Sub-functions (technical solutions)	Base frame	Cables	Inner light	Breaking light	"padding"	Rear clip	Instep handle	Sun shield	Bearings	Mirror	Reading light	Roof hatch conn.	**Score**
Carry over			●	●		●	●	●	●	●	●	●	81
Technology push		●			◍								12
Planned Design changes													0
Technical Specification	◍		○		○			◍		○			9
Styling	●												9
Common unit		●	◍	◍	○	●	●	○	●	◍	●	●	65
Process / Oragnisation							●	●		●			27
Separate testing													0
black box		●	●	●				●			●		45
Service / Maintenance	◍												3
Upgrading													0
Recycling	●	●	●	●				●			●		54
Score	24	36	31	30	5	18	27	40	18	22	36	18	

Figure 4: MIM for the modules of the inner roof of a vehicle (Erixon 1998)

Module Drivers can be used to indicate and justify the strategic reasons for creating modules. Three categories of value in which all Module Drivers fall are Product Leadership, Operational Excellence, and the Customer Intimacy. Each category represents a strategic reasoning behind the modularization of each component. To create modules from MFD, technical solutions, or product components, are compared to identify groups with similar Module Drivers in the MIM. This may be done manually or through the use of hierarchical clustering. Hierarchical clustering evaluates the closeness between the scoring of components' Module Drivers and clusters the components accordingly.

Hölttä and Salonen (2003) carried out a comparison of the three described modularity methods applied to four commercial products. The study found that each method, given the same inputs, structures the architectures of the products differently. This is because the methods differ in the objectives of their clustering. Function structure heuristics focus more on functionality, whereas DSM focuses on component interactions and MFD is based on strategic reasoning behind modularization. The study also found that, out of the three methods, only function structure heuristics are fit to apply directly to modularizing product families. To integrate both component interactions and company strategy in the module generation process, Borjesson and Hölttä-Otto (2014) proposed a DSM clustering algorithm that considers MFD module drivers in its clustering. This method considers both component interactions and strategic reasoning in the module clustering process.

2.2.3 Methods for Designing Modular Product Families

A modular product family consists of one or more modules shared between products in the family. In designing modular product families, designers must consider both the strategic modular architecture for each product and the choice of shared modules for the product platform.

Several methods focus on the use of heuristics to identify modules from the function structures of product families. Zamirowski and Otto (1999) build on the single product function structure heuristics. In their method, they present an approach to take the function structures of each individual product and combine them into a product family function structure. In this paper, three variant heuristics for product families are proposed. These are included with the original three heuristics for use in modularizing the product family function structure and identifying shared and unique modules. Dahmus et al. (2001) also make use of the product family function structure to identify shared and unique functions in a product family. This method organizes each shared and unique function into a function versus product matrix and then clusters functions into modules based on the modularity heuristics. Sudjianto and Otto (2001) introduced a similar matrix-based method that focuses on platforms designed around nontechnical aspects, such as shape and color. Hölttä-Otto et al. (2008) also used product function structures to identify common modules. In this approach, the commonality of functional modules is identified and quantified. Commonality is calculated using the Euclidian distance between the functions' input and output flows.

Several publications focus on the modularization of product families with predefined sets of components. Rojas Arciniegas and Kim (2011) proposed a method to define product family modules at the detailed design stage. Components are first mapped to functions to determine components or groups of components with the same functions. The components of each product are then clustered into modules using a metric of the impact to change a component in the platform and DSM clustering. This approach produces an optimal solution for clustering and sharing components based on the metrics used. Hsia and Liu (2005) developed an approach that organizes predefined product components into shared and variant modules. This approach uses Quality Function Deployment (QFD) to identify drivers for variation and the Interpretive Structural Model (ISM) to show the interactions of components. Otto et al. (2013) advocated an approach for using DSM to cluster the components of a product family into modules. However, this approach required that each product in the product family be highly similar and contain the same general set of components. Using this approach, modules were identified that each product in the family possessed, after which the sharing of each module between products was determined.

2.3 Methodology

In an effort to extend these predominant component-based modularity methods to product family design, the following method was developed. The following presents an algorithmic approach to identifying potential product family architectures that possess both high commonality and optimal module boundaries. The aim of this method is to allow designers to quickly produce a number of different architectures from which they may choose the best compromise of strategic modular design and commonality for their specific application.

In the proposed method, the components of a product family are divided into groups based on the set of products between which they may be shared. The individual groups are then clustered into modules using DSM clustering and combined to form the complete modular architecture of each product. Further alternative modular architectures are considered by strategically decreasing the amount of sharing in the product family to move common components to groups of less commonality which contain components with which they are highly coupled. This process follows a trade-off, creating better modules at the cost of commonality in the product line.

To evaluate alternatives, each proposed architecture is scored based on commonality, the quality of the modular architecture, and the minimum number of building blocks needed to construct the product family. Taking advantage of algorithmically based DSM clustering, a Pareto front of maximum commonality and optimal modularity can be computationally determined. This front provides modular architecture alternatives from which the final design of the product family may be chosen or further refined.

2.3.1 Method Input

The proposed method requires two primary inputs: (1) a list of the components that make up the product and (2) a DSM matrix for the product family.

The list of products and their constituent components may be encapsulated in a binary matrix, termed the Product-Component Matrix (PCM). Each row of the PCM corresponds to a product in the product family. Each column corresponds to a component used in at least one of the products.

$$\begin{Bmatrix} product\ 1 \\ \cdots \\ product\ m \end{Bmatrix} = \begin{matrix} \mathbf{PCM} \\ [m \times n\ binary\ matrix\ ] \end{matrix} \begin{Bmatrix} component\ 1 \\ \cdots \\ component\ n \end{Bmatrix} \tag{1}$$

Determining what components each product may share is an important part of the setup of this approach. There exist a number of different product family optimization approaches that focus on determining component sharing based on performance objectives. A component may be shared if the component is functionally, morphologically, and parametrically the same in each product. Decisions on component sharing that affects product performance should be made prior to the construction of the PCM.

2.3.2 Module Clustering

To begin the module clustering process, the components in the PCM are first organized into groups based on the set of products to which each component is common. This grouping of components is designed around one rule: if Component A and Component B are both included in the same set of products, then these components are placed into the same group.

These potential groups are termed shared groups. They can be visualized using a Venn diagram. Each segment of the diagram contains components that are shared by the same set of products, different than that of components in other groups. The Venn diagram in Figure 5 shows the component groups possible for a three-product product family.

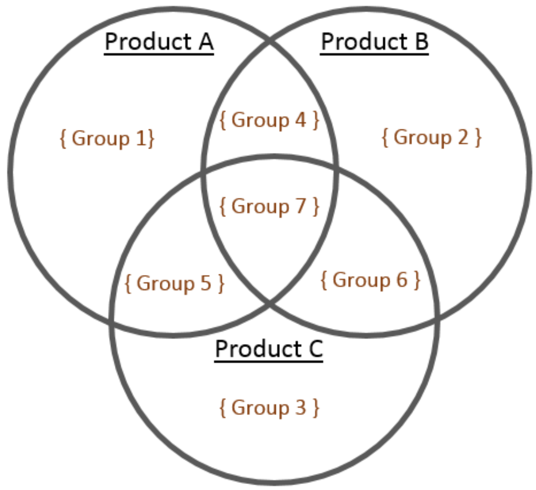


Figure 5: Shared groups of a three-product product family

After organizing the components into groups based on sharing, the DSM for the product family is used to cluster each shared group into modules individually. For each group, a new DSM is created from the DSM of the product family. The new DSM for each given group contains only the components within that shared group and their interactions with each other. Clustering is then conducted using a DSM clustering algorithm. This approach modularizes each of the groups separately. Modularizing the product family in this manner produces a modular architecture with the maximum possible amount of commonality, given the PCM input.

2.3.3 Further Alternatives

Further modular architectures may be considered by decreasing the amount of commonality in the product family. A component that might be shared by a large number of products, could be shared by a lesser number to allow the component to be integrally modularized with another group. An example of this is depicted in Figure 6.

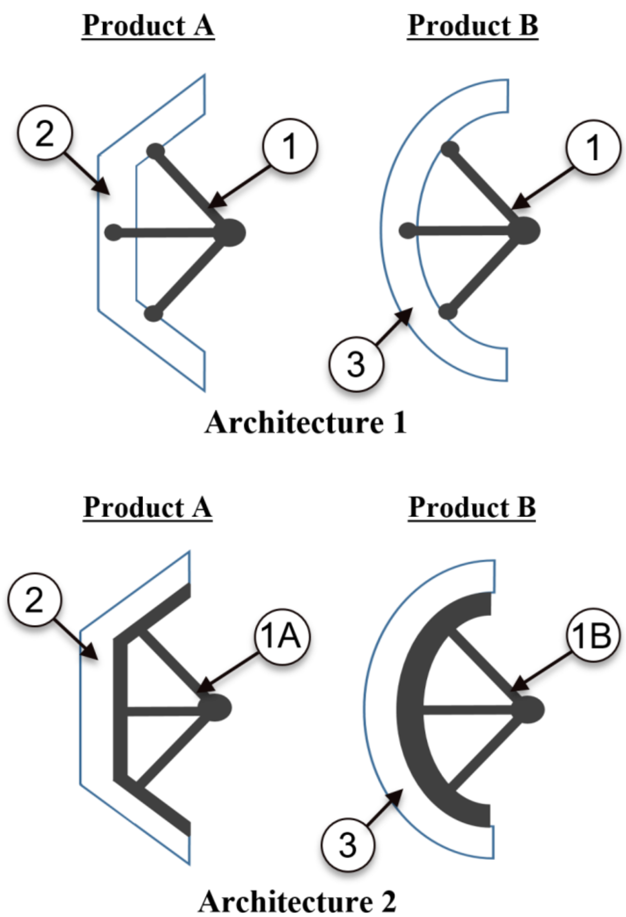


Figure 6: Alternative modular architectures for an example product family (adapted from Robertson and Ulrich 1998)

In this example, the product family consists of two products. Product A and B both possess Component 1. However, Product A utilizes Component 2, whereas Product B contains Component 3. If the product family was designed for maximum commonality, Component 1 would be designed as a shared module, which could attach to both Component 2 and Component 3. In this instance, Component 1 is common to both products. Components 2 and 3 are distinct to Product A and B, respectively. However, if Component 1 has high connectivity with both of the other two components, another alternative architecture could be considered in which Component 1 is designed into two separate modules: one in which it is combined with Component 2 and another combined with Component 3. In this example, a component that could potentially be shared between products is chosen not to be shared in order to create a better modular architecture. Thus, at the cost of commonality, the modular architecture is improved.

This process of decreasing commonality in order to attain different possibilities for modularizing the product can be represented again using a Venn diagram. The diagram in Figure 7 demonstrates the differences in the two architectures from the previous example.

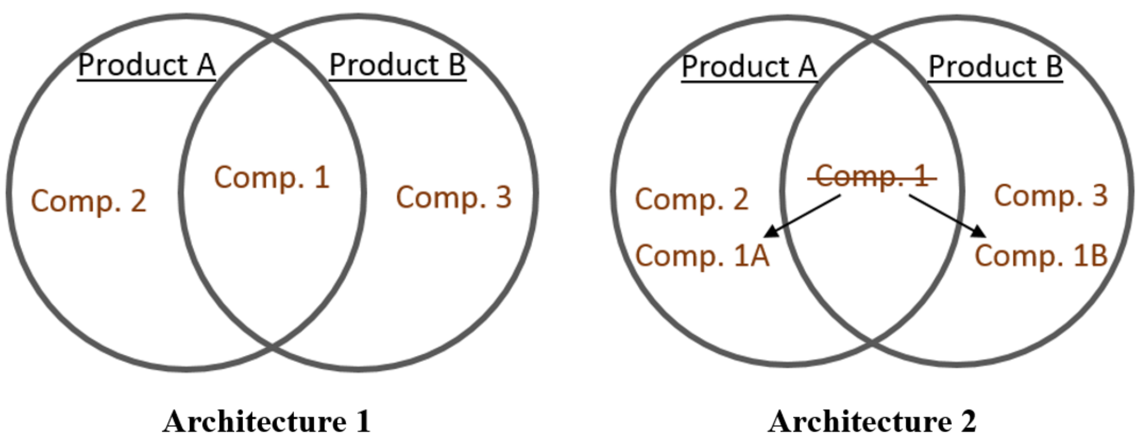


Figure 7: Venn diagrams depicting example Architectures 1 and 2

In the example, Component 1 could have been designed common to both products. However, to attain a different modular architecture, the component was designed differently between the two products. Thus, two distinct instances of Component 1 were created. The figure above shows this occurrence. In Architecture 1, Component 1 was located in the center of the Venn diagram, shared by both Product A and B. However, in the design of Architecture 2, two instances were created: Component 1A, which is unique to Product A and Component 1B, which is unique to Product B. Both products still possess the component, but it is no longer a shared component. Following the clustering rule defined in the previous section, moving the component into different segments of the Venn diagram, or different shared groups, allows the product to be modularized differently.

This process, of decreasing sharing to obtain different modular architectures can be carried out for any component in the product family. Figure 8 below demonstrates how a component could be divided into multiple instances to allow for modularization with different groups of components (letters represent instances of the same component).

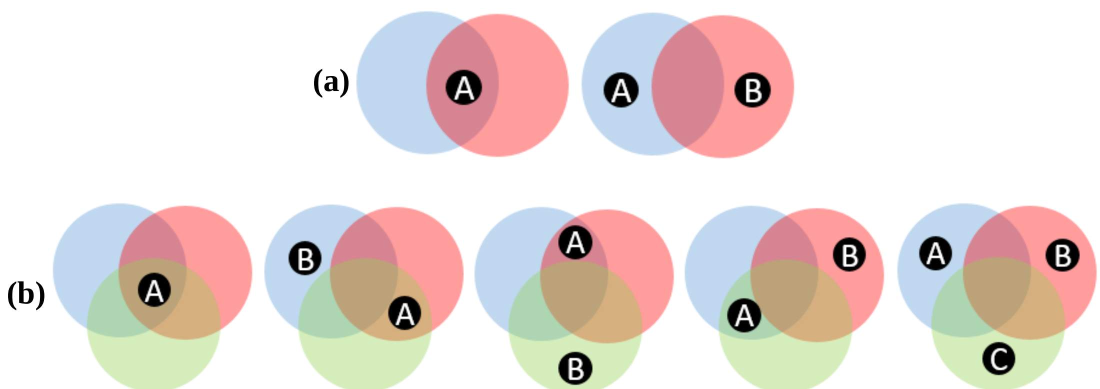


Figure 8: Possibilities for sharing a component common to (a) two products and (b) three products

Repeatedly carrying out this process for each of the components in the product family ultimately leads to a product family with no common components. In this case, each product in the product family may be modularized independently. Thus, the optimal modular architecture for each product may be chosen.

2.3.4 Number of Alternatives

Because of the computational nature of the described method, it is valuable to determine the number of operations required to analyze the product family. As the number of products that contain a given component increases, the number of possible configurations for that component increases with its corresponding Bell number (Bell 1934), B_{NP}. Bell numbers give the number of ways a set of elements can be partitioned into nonempty subsets, or in this case, the number of ways a component can be split into component instances shared by different sets of products. Table 1 gives the first nine Bell numbers.

Table 1: List of Bell numbers

n	1	2	3	4	5	6	7	8	9	…
B_n	1	2	5	15	52	203	877	4140	21147	…

Given that each component that is common to *n* number of products may be shared in B_n number of ways, the maximum number of alternatives for sharing the components of a product family can be calculated using the equation below.

$$Number\ of\ Alternatives = \prod_i^C B_{p_i} \tag{2}$$

In this equation, B_{p_i} is the Bell number for p_i, p_i is the number of products that can share component i, and C is the number of components in the product family.

To decrease the computation time of finding optimal designs, strategic search techniques can be employed. We can improve the search for Pareto-optimal architectures, by utilizing the DSM of the product family to predetermine what shared groups components should be moved to in order to improve the modularity score of the product family. In this case, a component would only be moved into a lower level of sharing that contains component(s) with which it has DSM connections, thus eliminating a large number of alternatives from evaluation.

2.3.5 Evaluating Alternatives

To evaluate the large number of alternatives for modularizing the product family, a commonality, modular architecture, and minimum building block score are devised. While the following metrics provide a good basis for assessing a proposed architecture, these metrics may be altered or added upon to best meet design-specific goals.

2.3.5.1 Assessing Commonality

A number of different indices have been developed for assessing commonality within a product family. In the proposed method, the Total Constant Commonality Index (TCCI) (Wacker and Trelevan 1986) is utilized to quantify the amount of commonality in a given architecture. The TCCI assesses commonality based on the number of parents of each component in the family. The equation for calculating the TCCI is given below.

$$TCCI = 1 - \frac{d-1}{\sum_{j}^{d} \Phi_j - 1} \tag{3}$$

In this equation, d is the number of distinct component instances, and Φ_j is the number of products to which component instance j is common.

The TCCI was chosen for its ease of calculation and setup (Thevenot and Simpson 2006). The TCCI provides a good initial estimate of the benefits of added commonality; however, it provides a relatively simplified view of commonality. The TCCI weights each component equally when assigning the commonality score. Therefore, it may not fully account for the complex benefits gained from sharing each component. A more detailed representation of the benefits of commonality can be attained using an index that requires more setup. To assess the commonality of each architecture alternative, designers may choose whichever index is best suited for their application.

2.3.5.2 Assessing Modular Architecture

To assess the quality of a proposed modular architecture, a DSM cost function, used in DSM clustering algorithms, can be used. In DSM clustering algorithms, a cost is calculated for each proposed clustering of components into modules. This clustering cost is a sum of IntraClusterCost, or cost of interactions occurring within a cluster, and ExtraClusterCost, or cost from interactions occurring outside of any cluster. DSM clustering algorithms attempt to minimize this cost function in their search for the optimal clustering, or modularization, of a product. Using the cost function of a chosen DSM clustering algorithm, modular architectures for the product family can be compared.

Ideally, to obtain the optimal modular architecture for a given product, the product would be clustered into modules without consideration of other products in the family. However, designing modules to be common to multiple products adds restrictions to how the products are modularized. The minimum module clustering cost of a given product is obtained when each product DSM is clustered without consideration of commonality.

Knowing this, the clustering cost of a sub-optimally clustered product, modularized as part of a product family, can be compared with the minimum possible clustering cost of that product. Comparing with the minimum clustering cost allows any proposed clustering of a product to be evaluated.

An example product is used to demonstrate scoring of the modular architecture of a product family. First, a DSM matrix for the product is created. The binary DSM matrix for the example product is shown in Figure 9.

	1	2	3	4	5	6	7	8	9	10
1		X	X							
2	X		X							
3	X	X							X	
4					X					
5			X	X					X	
6							X			
7						X		X		
8						X	X			
9			X		X					X
10									X	

Figure 9: Optimally clustered DSM of the example product

Figure 9 shows the example product clustered into modules. In this example, the product consists of only 10 components. The DSM clusters the product into 4 modules ({1 2 3}, {4 5}, {6 7 8}, and {9 10}). This is the optimal clustering of the product, given the DSM input. Thus, the optimal DSM clustering cost for this product is 71.

In the previous sections, alternatives for modularizing the components in the product family are suggested by organizing the components into shared groups. On such architecture might sort the product into three groups ({1 2 4}, {3 8 9}, {5 6 7 10}).

To evaluate quality of the modular architecture of this design alternative, each group is first individually clustered to identify modules using the DSM clustering algorithm. The optimal clustering of each of these groups is given below in Figure 10.

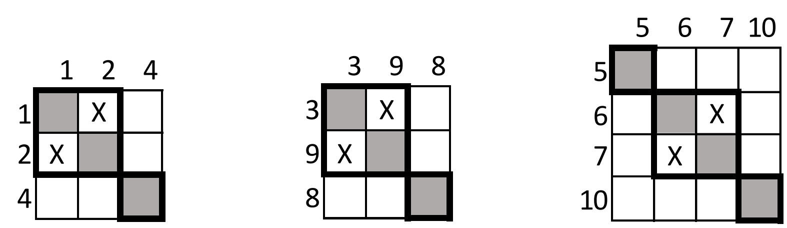


Figure 10: Three shared groups of the example product with individual DSM clustering

Figure 10 shows the result of the clustering for each individual component group. These modules are then combined back into one matrix, and the DSM score for the resulting clustered matrix is calculated. Figure 11 shows the clustering of the product obtained from recombining each individually clustered shared group.

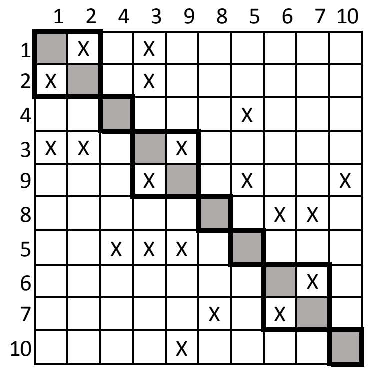


Figure 11: Final clustered DSM Matrix for the example product from the recombined shared groups

The final, regrouped matrix has a significantly larger clustering cost of 140. This cost is compared to the optimal DSM score to calculate the modularity score for this product in the product family.

$$Modularity\ Score\ for\ Product\ i = MS_i = \frac{DSM\ Clustering\ Cost_{optimal,i}}{DSM\ Clustering\ Cost_{proposed,i}} \times 100 \quad (4)$$

$$Example\!: \ \ MS_1 = \frac{71}{140} \times 100 = 50.7$$

Using this technique, a score can be assigned to the modular clustering of each product in the family. By weighting each product, a single modularity score for the product family can be determined. Weighting each product also allows more flexibility in and control over the design process, as products deemed to be more important to the success of the product line may be weighted more heavily than other products. Assuming the product in this example is part of a family of two other products with modularity scores of 33.0 and 60.0 ($MS_2 = 33.0, MS_3 = 60.0$) and that each product is equally weighted ($w_1 = w_2 = w_3 = 1/3$), the modularity score for the product family is calculated in the equation below.

$$Modularity\ Score\ for\ Product\ Family = MS_{PF} = \sum_i^N w_i \cdot MS_i \tag{5}$$

$$Example\text{:}\ \ MS_{PF} = \frac{50.7 + 33.0 + 60.0}{3} = 47.9$$

The modularity score calculated using this method can then be used to compare different modular architecture alternatives. To obtain modular product family architectures with higher modularity scores, commonality must often be decreased. Thus, various options for architecting the product family may be plotted as a Pareto front of maximized commonality and modularity score.

2.3.5.3 Assessing Minimum Number of Building Blocks

In many cases, modularity may be used in a product family to organize the products into building block modules, which facilitate assembly and configurability. In this case, the goal of designing the product family may be to minimize the number of modules required to assembly the family. The minimum number of building block modules may be calculated by utilizing a DSM that represents the physical connections between components. This connectivity DSM may be the same as that used to complete the primary clustering of modules, or could be included in addition.

After organizing the product family into modules based on the clustering DSM, two or more of these modules may be combined to form building blocks. Within each shared group in the product family, the identified modules are subsequently analyzed using the connectivity DSM to determine if the components of each module possess connections with components in other modules which allow them to be combined into a building block. By combining all modules in each shared group that are interconnected, the minimum number of building block modules needed to assemble the product family can be calculated.

2.4 Case Study

To demonstrate the usefulness of the proposed method, it is applied to the modularization of a family of high- and low-end impact drivers and electric drills. The product family includes: (1) a low-end electric drill, (2) a low-end impact driver, (3) a high-end brushless electric drill, and (4) a high-end brushless impact driver. The list of products and components in the product family are given in Table 2 and Table 3. The PCM for the product family is included in Appendix A.

Table 2: Product list for the power tool product family case study

Product No.	Product Description
1	Low-End Electric Drill
2	Low-End Impact Driver
3	High-End Electric Drill
4	High-End Impact Driver

Table 3: Component list for the power tool product family case study

Component No.	Component Description	Component No.	Component Description
1	Clamshell	21	VSR Switch HE
2	Armature 1	22	Heat Sink
3	Armature 2	23	Electronics Board
4	Stator Magnet 1	24	Belt Clip
5	Stator Magnet 2	25	Bit Clip
6	Motor Brushes	26	Drill Light
7	Brush Holders	27	Impact Driver Light
8	Commutator	28	Chuck HE
9	Rear Bearing	29	Chuck LE
10	Pinion Gear	30	Transmission LE
11	Front Bearing	31	Transmission HE
12	Motor Fan	32	Impact Mech LE
13	Motor ESC	33	Impact Mech HE
14	Permanent Magnet 1	34	Anvil LE
15	Permanent Magnet 2	35	Anvil HE
16	Stator 1	36	Nose Cone
17	Stator 2	37	Battery Terminal
18	Trigger	38	20V Battery
19	Fwd/Rev Switch	39	Grip LE
20	VSR Switch LE	40	Grip HE

The interactions between components in each product were represented in a single binary DSM. Algorithmic DSM clustering and the calculation of clustering costs was completed using the clustering algorithm presented by Thebeau (2001). From the product family DSM, each individual product's DSM is created. The DSM for the product family is given in Figure 12. The optimally clustered DSMs for each product variant are included in the Appendix A.

	Clamshell 1	Armature 1 2	Armature 2 3	Stator Magnet 1 4	Stator Magnet 2 5	Motor Brushes 6	Brush Holders 7	Commutator 8	Rear Bearing 9	Pinion Gear 10	Front Bearing 11	Motor Fan 12	ESC 13	Magnetic Rotor 1 14	Magnetic Rotor 2 15	Stator 1 16	Stator 2 17	Trigger 18	Fwd/Rev Switch 19	VSR Switch LE 20	VSR Switch HE 21	Heat Sink 22	Electronics Board 23	Belt Clip 24	Bit Clip 25	Drill Light 26	Impact Driver Light 27	Chuck HE 28	Chuck LE 29	Transmission 1 30	Transmission 2 31	Impact Mech LE 32	Impact Mech HE 33	Anvil LE 34	Anvil HE 35	Nose Cone 36	Battery Connector 37	20V Battery 38	Grip LE 39	Grip HE 40
Clamshell 1	X																																						X	X
Armature 1 2		X						X	X	X	X																													
Armature 2 3			X					X	X	X	X																													
Stator Magnet 1 4		X		X																																				
Stator Magnet 2 5			X		X																																			
Motor Brushes 6						X	X	X																																
Brush Holders 7	X					X	X																																	
Commutator 8		X	X					X																																
Rear Bearing 9	X	X	X						X					X	X																									
Pinion Gear 10		X	X							X				X	X															X	X	X	X							
Front Bearing 11	X	X	X								X			X	X																									
Motor Fan 12		X	X									X		X	X																									
ESC 13													X			X	X																							
Magnetic Rotor 1 14									X	X	X			X		X																								
Magnetic Rotor 2 15									X	X	X				X		X																							
Stator 1 16	X												X	X		X																								
Stator 2 17	X												X		X		X																							
Trigger 18																		X		X	X																			
Fwd/Rev Switch 19																			X	X	X																			
VSR Switch LE 20																		X	X	X			X														X			
VSR Switch HE 21																		X	X		X		X																	
Heat Sink 22																						X	X																	
Electronics Board 23													X							X	X	X	X														X			
Belt Clip 24	X																							X																
Bit Clip 25	X																								X															
Drill Light 26	X																						X			X														
Impact Driver Light 27																				X			X				X									X				
Chuck HE 28																												X												
Chuck LE 29																													X											
Transmission 1 30																													X	X										
Transmission 2 31																												X			X									
Impact Mech LE 32																																X		X		X				
Impact Mech HE 33																																	X		X	X				
Anvil LE 34																																		X						
Anvil HE 35																																			X					
Nose Cone 36	X																										X					X	X	X	X	X				
Battery Connector 37	X																			X			X														X			
20V Battery 38																																						X		
Grip LE 39	X																																						X	
Grip HE 40	X																																							X

Figure 12: DSM of the case study power tool product family

Using the proposed method and the PCM and DSM inputs for the product family, Pareto fronts of architectures with maximum commonality and modularity score were identified. Each front corresponds to a number of building blocks needed to construct the family. The Pareto front was identified, starting from the maximum commonality architecture. To search for Pareto optimal architectures, components of high commonality were iteratively moved into shared groups of lower levels of commonality that contain components with which they possess DSM interactions. Architectures that demonstrated either higher modularity score or lower number of building blocks were kept, while those that did not were discarded. The DSM in Figure 12 was used as the connectivity DSM when calculating the number of building blocks needed for assembly. Figure 13 shows the Pareto fronts produced and where along the front the chosen architecture lies.

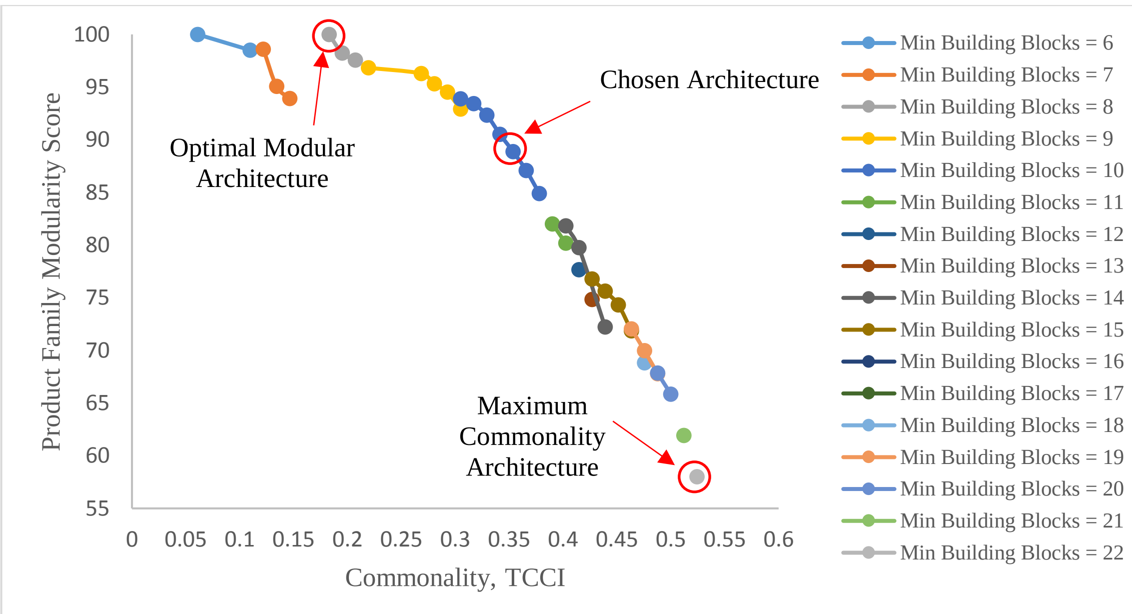
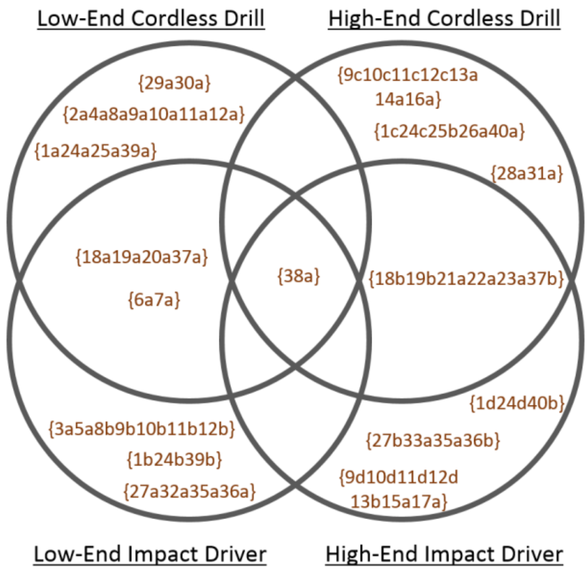


Figure 13: Pareto front of modular architectures with maximized commonality and optimal modular architecture for the power tool case study

From the Pareto front, three architectures are chosen for further analysis. The architectures with optimal modularity and maximum commonality are chosen to demonstrate the extremes of the front. An architecture with a compromise of these metrics is chosen to exemplify a design that might be chosen from the front. Table 4, Table 5, and Table 6 present the modular architecture design and corresponding clustering costs for the optimal modularity architecture, maximum commonality architecture, and chosen design architecture, respectively. Figure 14, Figure 15, and Figure 16 also demonstrate the sharing of modules within the family and the relevant assessment metrics for each respective architecture.

Table 4: Optimal modular architecture of each product in the power tool product family

Product	Min Clustering Cost	Modules	Module Description
Low-End Cordless Drill	236	{1, 24, 25, 39}, {2, 4, 8, 9, 10, 11, 12}, {6, 7}, {18, 19, 20, 37}, {29, 30}, {38}	{Clamshell, Belt Clip, Bit Clip, Grip LE}, {Armature 1, Stator Magnet 1, Commutator, Rear Bearing, Pinion Gear, Front Bearing, Motor Fan}, {Motor Brushes, Brush Holder}, {Trigger, Fwd/Rev Switch, VSR Switch LE, Battery Terminal}, {Chuck LE, Transmission LE}, {20V Battery}
Low-End Impact Driver	299	{1, 24, 39}, {3, 5, 8, 9, 10, 11, 12}, {6, 7}, {18, 19, 20, 37}, {27, 32, 34, 36}, {38}	{Clamshell, Belt Clip, Grip LE}, {Armature 2, Stator Magnet 2, Commutator, Rear Bearing, Pinion Gear, Front Bearing, Motor Fan}, {Motor Brushes, Brush Holder}, {Trigger, Fwd/Rev Switch, VSR Switch LE, Battery Terminal}, {Impact Driver Light, Impact Mech LE, Anvil LE, Nose Cone}, {20V Battery}
High-End Cordless Drill	311	{1, 24, 25, 26, 40}, {9, 10, 11, 12, 13, 14, 16}, {18, 19, 21, 22, 23, 37}, {29, 31}, {38}	{Clamshell, Belt Clip, Bit Clip, Drill Light, Grip HE}, {Rear Bearing, Pinion Gear, Front Bearing, Motor Fan, Motor ESC, Magnetic Rotor 1, Stator 1}, {Trigger, Fwd/Rev Switch, VSR Switch HE, Heat Sink, Electronics Board, Battery Terminal}, {Chuck HE, Transmission HE}, {20V Battery}
High-End Impact Driver	338	{1, 24, 40}, {9, 10, 11, 12, 13, 15, 17}, {18, 19, 21, 22, 23, 37}, {27, 33, 35, 36}, {38}	{Clamshell, Belt Clip, Grip HE}, {Rear Bearing, Pinion Gear, Front Bearing, Motor Fan, Motor ESC, Magnetic Rotor 2, Stator 2}, {Trigger, Fwd/Rev Switch, VSR Switch HE, Heat Sink, Electronics Board, Battery Terminal}, {Impact Driver Light, Impact Mech HE, Anvil HE, Nose Cone}, {20V Battery}



Commonality Score: 0.183

Minimum Building Blocks: 8

Modular Architecture Score: 100

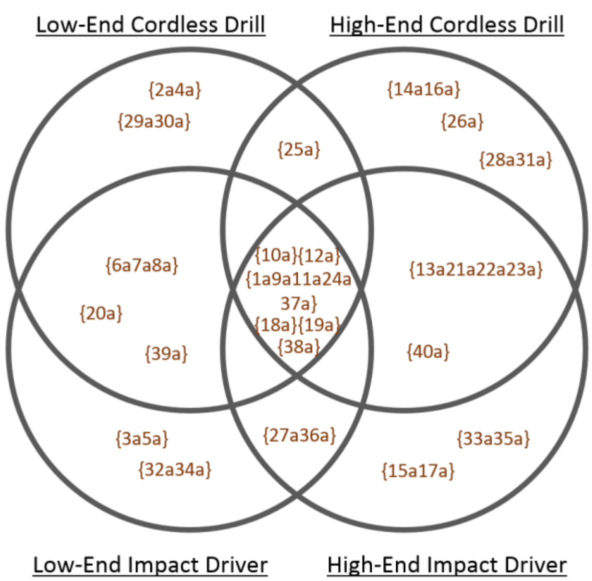
Figure 14: Module sharing and assessment metrics for the architecture with optimal modularity

Table 5: Modular architecture of power tool product family with maximum commonality

Product	Clustering Cost	Modules	Module Description
Low-End Cordless Drill	433	{1, 9, 11, 24, 37}, {2, 4}, {6, 7, 8}, {10}, {12}, {18}, {19}, {20}, {24}, {29, 30}, {38}, {39}	{Clamshell, Rear Bearing, Front Bearing, Belt Clip, Battery Terminal}, {Armature 1, Stator Magnet 1}, {Motor Brushes, Brush Holder, Commutator}, {Pinion Gear}, {Motor Fan}, {Trigger}, {Fwd/Rev Switch}, {VSR Switch LE}, {Bit Clip}, {Chuck LE, Transmission LE}, {20V Battery}, {Grip LE}
Low-End Impact Driver	541	{1, 9, 11, 24, 37}, {3, 5}, {6, 7, 8}, {10}, {12}, {18}, {19}, {20}, {27, 36}, {32, 34}, {38}, {39}	{Clamshell, Rear Bearing, Front Bearing, Belt Clip, Battery Terminal}, {Armature 2, Stator Magnet 2}, {Motor Brushes, Brush Holder, Commutator}, {Pinion Gear}, {Motor Fan}, {Trigger}, {Fwd/Rev Switch}, {VSR Switch LE}, {Impact Mech LE, Anvil LE}, {Impact Driver Light, Nose Cone}, {20V Battery}, {Grip LE}

Table 5: (Continued)

Product	Clustering Cost	Modules	Module Description
High-End Cordless Drill	508	{1, 9, 11, 24, 37}, {10}, {12}, {13, 21, 22, 23}, {14, 16}, {18}, {19}, {25}, {26}, {28, 31}, {38}, {40}	{Clamshell, Rear Bearing, Front Bearing, Belt Clip, Battery Terminal}, {Pinion Gear}, {Motor Fan}, {Motor ESC, VSR Switch HE, Heat Sink, Electronics Board}, {Magnetic Rotor 1, Stator 1}, {Trigger}, {Fwd/Rev Switch}, {Bit Clip}, {Drill Light}, {Chuck HE, Transmission HE}, {20V Battery}, {Grip HE}
High-End Impact Driver	554	{1, 9, 11, 24, 37}, {10}, {12}, {13, 21, 22, 23}, {15, 17}, {18}, {19}, {27, 36}, {33, 35}, {38}, {40}	{Clamshell, Rear Bearing, Front Bearing, Belt Clip, Battery Terminal}, {Pinion Gear}, {Motor Fan}, {Motor ESC, VSR Switch HE, Heat Sink, Electronics Board}, {Magnetic Rotor 2, Stator 2}, {Trigger}, {Fwd/Rev Switch}, {Impact Driver Light, Nose Cone}, {Impact Mech HE, Anvil HE}, {20V Battery}, {Grip HE}



Commonality Score: 0.524

Minimum Building Blocks: 22

Modular Architecture Score: 58.0

Figure 15: Module sharing and assessment metrics for the architecture with maximum commonality

Table 6: Chosen modular architecture for the power tool product family

Product	Clustering Cost	Modules	Module Description
Heavy-Duty Impact Driver	271	{1, 11, 24, 39}, {2, 4, 9, 10}, {6, 7, 8}, {12}, {18, 19, 20, 37}, {25}, {29, 30}, {38}	{Clamshell, Front Bearing, Belt Clip, Grip LE}, {Armature 1, Stator Magnet 1, Rear Bearing, Pinion Gear}, {Motor Brushes, Brush Holder, Commutator}, {Motor Fan}, {Trigger, Fwd/Rev Switch, VSR Switch LE, Battery Terminal}, {Bit Clip}, {Chuck LE, Transmission LE}, {20V Battery}
Lightweight Impact Driver	324	{1, 11, 24, 39}, {3, 5, 9, 10}, {6, 7, 8}, {12}, {18, 19, 20, 37}, {27, 32, 34, 36}, {38}	{Clamshell, Front Bearing, Belt Clip, Grip LE}, {Armature 2, Stator Magnet 2, Rear Bearing, Pinion Gear}, {Motor Brushes, Brush Holder, Commutator}, {Motor Fan}, {Trigger, Fwd/Rev Switch, VSR Switch LE, Battery Terminal}, {Impact Mech LE, Impact Driver Light, Nose Cone, Anvil LE}, {20V Battery}
Heavy-Duty Drill	368	{1, 11, 24, 40}, {9, 10, 14, 16}, {12}, {13, 22, 23, 37}, {18, 19, 21}, {25}, {26}, {28, 31}, {38}	{Clamshell, Front Bearing, Belt Clip, Grip HE}, {Rear Bearing, Pinion Gear, Magnetic Rotor 1, Stator 1}, {Motor Fan}, {Motor ESC, Heat Sink, Electronics Board, Battery Terminal}, {Trigger, Fwd/Rev Switch, VSR Switch HE}, {Bit Clip}, {Drill Light}, {Chuck HE, Transmission HE}, {20V Battery}
Lightweight Drill	369	{1, 11, 24, 40}, {9, 10, 15, 17}, {12}, {13, 22, 23, 37}, {18, 19, 21}, {27, 33, 35, 36}, {38}	{Clamshell, Front Bearing, Belt Clip, Grip HE}, {Rear Bearing, Pinion Gear, Magnetic Rotor 2, Stator 2}, {Motor Fan}, {Motor ESC, Heat Sink, Electronics Board, Battery Terminal}, {Trigger, Fwd/Rev Switch, VSR Switch HE}, {Impact Driver Light, {Impact Mech HE, Anvil HE, Nose Cone}, {20V Battery}

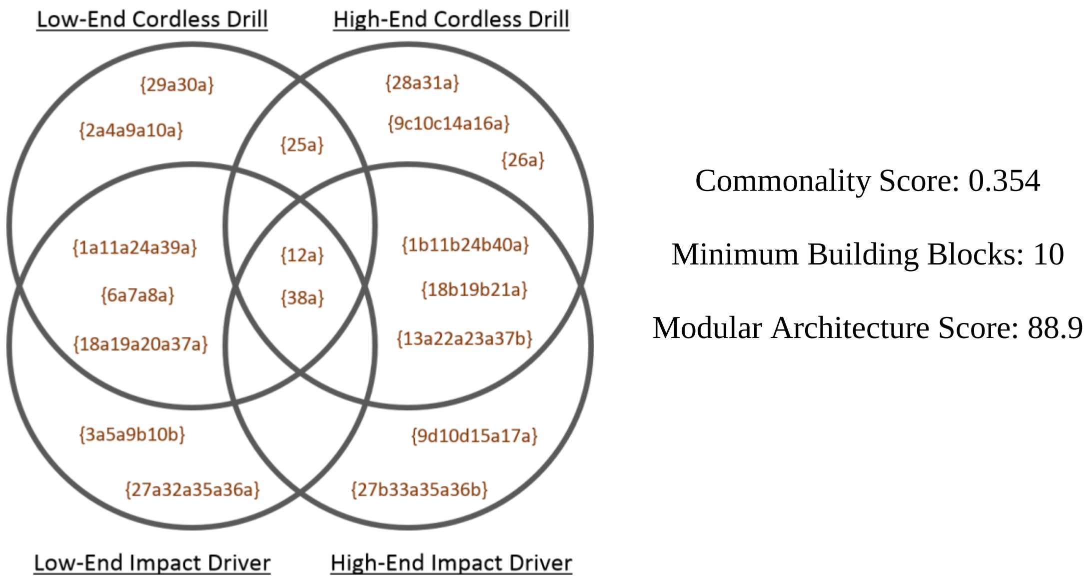


Figure 16: Module sharing and assessment metrics for the chosen architecture

The architecture with the optimal modularity score has a modularity score of 100. In this architecture, each product is individually clustered to produce the minimum possible clustering cost. While this clustering provides the ideal clustering of each product, few of the modules may be shared between products in the family resulting in a commonality of 0.183.

In the architecture with maximum commonality, every component that can be shared between products in the family is shared, resulting in a commonality of 0.524. However, this commonality adds restrictions to the clustering that result in higher clustering costs and a modularity score of only 58.0. This architecture also results in a large number of small modules that cannot be combined into larger building blocks.

The chosen design provides a compromise of commonality and high modularity score. This architecture has a commonality of 0.354 and a modularity score of 88.9. This architecture also requires only 10 building blocks to construct, much less than that required by the maximum commonality architecture.

2.5 Conclusion and Future Work

This article introduces a new technique for architecting modular product families with a balance of both high commonality and strategic modular architecture. The proposed method allows designers to quickly and easily identify a variety of modular architectures that lie along a Pareto front of maximum commonality and strategic modularity. From this front, alternatives can be evaluated, compared, and further modified. This method offers more alternatives for the sharing of components than other previously developed methods. This method also helps designers address the uncertainty in the design process by providing a multitude of design options from which they can utilize area specific expertise that can often be difficult to quantify in an algorithmic approach. The method is robust in that it can be used for a number of different DSM clustering algorithms and commonality indices available. This method requires little setup, using common design practices, and can be implemented either computationally or manually.

The case study in this article was completed using simple, binary DSM clustering. However, the method could be adapted for use with other modularity methods that computationally organize components into modules. MFD, which often utilizes hierarchical clustering, could potentially be implemented in this method to compare modular architectures based on hierarchical clustering metrics. Many other DSM clustering algorithms also exist that could be used for module clustering. A hybrid DSM and MFD approach, such as that proposed by Borjesson and Hölttä-Otto (2014), could also be applied to method. The many different methods which can be utilized with this approach allow it to easily be adapted to fit the needs of a wide range of applications. However, because the method analyzes a large number of alternatives in the production of a Pareto front, it is limited by the time the chosen clustering method takes to find optimal module clusters. The method works well with simple DSM clustering because it is quickly completed; however, more computationally intensive clustering methods may require too long of a clustering process to timely carry out the analysis of various alternatives.

The method presented in this article also allows for ease of implementation with other common product family design methods. A primary research topic in product family is the parametric optimization of performance driving components. At this stage of the design process, the shared components are determined based on a tradeoff of commonality and performance. This step of the design process can be used to generate the PCM for the product family. Alternatively, if a performance metric were developed, it could be introduced and used in the iterative evaluation of architecture alternatives in this method. Even without such an iterative approach to product performance, the results of the method can be used to inform changes to improve the DSM and/or PCM of the product family. Figure 17 demonstrates the steps of designing a product family using this method, and how the output of the method can be used.

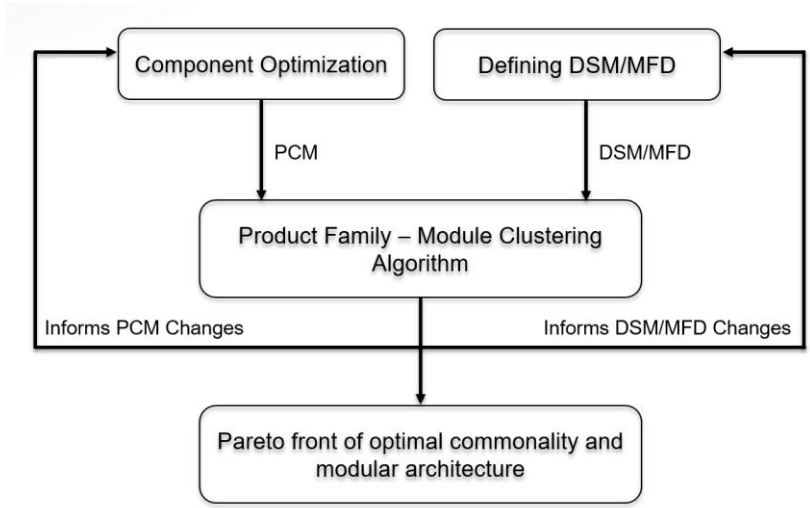


Figure 17: Work flow diagram of the inputs and output of the method

The method described in this article does have several limitations. As the size of the product family increases, the computational time greatly increases. For significantly large product families, the product family may need to be broken up and analyzed separately, determining the clustering of different sections of the product independently. To reduce the computation time, better search algorithms could also be implemented to more strategically select alternatives to analyze.

Ultimately, the case study and logic behind this paper show a tradeoff between commonality and the quality of the modular architecture in product family design. This paper suggests the optimization of the combination of performance and commonality of performance driving components before the implementation of the proposed module clustering algorithm. Thus, further decreases in commonality from the original PCM can primarily be attributed to the movement of auxiliary, non-performance driving components into shared groups that contain components with which they possess strong interactions. Using these findings, this technique can informally be utilized, even without the implementation of such an algorithmic approach. Modules in a product family can be identified by first deciding the commonality of performance driving components and then evaluating the optimal tradeoff between the commonality of auxiliary components and quality of the modular architecture, keeping in mind that commonality can be sacrificed to provide a better strategic modular architecture.

3 STRATEGY AND METHODS FOR THE DESIGN OF UNIVERSAL PRODUCT FAMILIES

3.1 Introduction

With the aging populations in many developed countries, an increasing number of individuals possess some level of disability (Lloyd-Sherlock 2000; United Nations 2015; Vincent and Velkoff 2010). Despite this, disabled users are often overlooked in the design process. To ensure that product designs meet the demands of all users and to capture the growing market of disabled users, cost-effective methods for designing more inclusive products are needed.

Universal product family design offers an economical option for creating accessible products for those with disabilities. Product family design is a proven method for adding variety to products (Simpson et al. 2006). A modular product family consists of shared modules (utilized by two or more products in the family) and unique modules (unique to an individual product). Using a combination of shared and unique modules, the functionality of the product can be interchanged. Designing typical and accessible products within modular product families allows the costs of shared elements to be leveraged across products while still meeting the needs of each group.

This paper details a proposed method for designing universal product families from functional modeling to detailed modular products. The method includes market segmentation, action-function modeling, detailed component design, and modularization.

This approach will be demonstrated using a case study of the design of a product family of typical and inclusive vehicle driver seats.

The structure of the paper is as follows: Section 3.2 presents the background to inclusive product family design, Section 3.3 details the design method proposed in this paper, Section 3.4 presents a case study of the design of typical and inclusive driver seats, and Section 3.5 includes conclusions drawn and potential future work.

3.2 Background

3.2.1 Universal/Inclusive Design

Universal design is a newly introduced term for the design of products that are fully usable by both regular and disabled customers (Mace 1985). The goal of universal design is to create designs that equally serve both fully able and disabled users simultaneously. Other terms used for universal design include: inclusive design, design for all, design for disability, and accessible design.

Researchers at North Carolina State University have developed seven principles that are key to universal design (Connell et al. 1997). These seven principles are: 1) equitable use, 2) flexibility in use, 3) simple and intuitive use, 4) perceptible information, 5) tolerance for error, 6) low physical effort, and 7) size and space for approach and use. Each of the seven principles has a distinct set of guidelines for meeting the design criteria. These principles provide guidance in the development of universal designs.

Older adults comprise one of the largest groups that often possess disabilities that limit their use of products. In 2010, about 13 percent of the United States population was 65 and older. It is estimated that, by the year 2030, this number will grow to 19 percent of the population (Vincent and Velkoff 2010). Prior research has sought to identify the distinct set of disabilities that affects elderly users. A guide to the concepts of universal design has been developed and applied to designs for the elderly (Story et al. 1998). Vanderheiden (1997) presented a set of guidelines for the universal design of consumer products and a list of human factors and ergonomics. Methods for measuring usability have been suggested, including usability studies and focus groups (Fisk et al. 2009).

Products that possess the same overall functionality but differ in their level of inclusiveness are termed a product pair (Kostovich et al. 2011). Figure 18 provides an example of a product pair of cutting utensils.

Figure 18: A product pair of a Fiskars Rotary Cutter (above) and a standard box cutter (below) (McAdams and Kostovich 2011)

Action-function diagrams were developed to identify differences between product pairs that improve product usability. (Kostovich et al. 2009). An action-function diagram is created by combining a product's activity diagram and functional model. Design differences identified using an action-function diagram can be classified into one of three categories: 1) parametric differences, 2) morphological differences, or 3) functional differences (McAdams and Kostovich 2011). Products with parametric differences possess the same general design but differ in the design parameters of the parts. A morphological difference is found in products with the same functionality but a different solution principle, form, or geometric topology. A functional difference indicates a change, deletion, or addition of a function of the product. To determine how successful universal designs are created, methods have been developed for identifying how typical product functions and user activities are made more inclusive through parametric, morphological, and/or functional charges (Sangelkar et al. 2012; Kostovich et al. 2009).

3.2.2 Product Family Design

Product family design, or product platform design, is one suggested method for the implementation of universal design. A product family consists of a group of products that share a common elements to satisfy a variety of market niches. Designing common components/modules to be shared by a group of similar products, the variant products in a product family can be efficiently developed (Meyer and Lehnerd 1997). Prior research has demonstrated product family design to be a cost-effective solution to provide variety in products and allow designers to meet the needs of multiple market segments (Simpson et al, 2006). A successful product family relies on the strategic selection of common elements that provide the most benefit without compromising the distinctiveness or performance of the individual products within the family.

Designing a product family requires the proper selection of shared components that results in high product commonality without sacrificed product distinctiveness. Typically, commonality provides economic benefits at the cost of product performance (Robertson and Ulrich 1998). Several commonality indices have been developed to assess the commonality of a product family. Thevenot and Simpson (2006) conducted a comparison of predominant commonality indices and provided recommendations for the use of index. Johnson and Kirchain (2010) assessed the correlation between a set of popular commonality indices and resultant cost savings from component sharing using process-based cost modeling.

To determine the variety needed from a product family, designers must first determine the different needs of customers that they are trying to reach with their product. The first step of product platform design is to identify and define market segments. Market segments are homogeneous groups of customer preferences (Meyer and Lehnerd 1997). Market segmentation helps designers plan platforms so that their product satisfies the needs of as many customers as possible with as few variations as possible. Hence, over-partitioning of the market may occur if market segmentations are clustered in a manner such that there are no major differences in customer needs between two or more segments. In product family design, common elements are leveraged across products that target multiple market segments. For the design of inclusive product families, markets are often segmented based on users' levels of impairment (Moon and McAdams 2012).

Modular product family design utilizes a combination of shared and unique modules to construct products. Ideally, modules contain high coupling internally and low external coupling with other modules. In the design of a universal product family, modules can be placed into one of three categories: universal, accessible, or typical modules (Moon and McAdams 2012). Universal modules are those shared by the products for both typical and disabled users. Accessible modules are those that provide usability to disabled users. Typical modules require usability functions that disabled users are not capable of completing. Defining the boundaries of the modules is a difficult problem in the design of modular products.

Several methods focus on the use of heuristics to identify modules from the function structures of product families. Zamirowski and Otto (1999) and Dahmus et al. (2001) proposed techniques for defining modules using the function structures of each individual product by combining them into a single product family function structure. Sudjianto and Otto (2001) introduced a similar matrix-based method that focuses on platforms designed around nontechnical aspects, such as shape and color. Hölttä-Otto et al. (2008) used product function structures to identify common modules base on the Euclidian distance between the input/output flows of each products' functions.

Several publications focus on the modularization of product families with predefined sets of components. Rojas Arciniegas and Kim (2011) proposed a method to define product family modules by identifying functionally similar components and utilizing DSM clustering. Hsia and Liu (2005) utilized Quality Function Deployment (QFD) to and an Interpretive Structural Model (ISM) to cluster modules. Otto et al. (2013) advocated an approach for using DSM to cluster the components of a product family into modules. However, this approach required that each product in the product family contain the same general set of components.

3.3 Methodology

The suggested approach begins with market segmentation to identify and target typical and impaired user groups. Action-function modeling of the typical product is utilized to determine needed design changes for adding usability. The design process is then carried out through concept development, component definition, and parametric optimization of the components to produce a list of components that compose the product family. The components of the product family are ultimately organized into shared and distinct modules and assembly building blocks.

3.3.1 Market Segmentation

The first step of the approach is to segment the market into the various groups which the specific product variants will target. In this stage, the customer base is divided into groups of users that have distinct preferences and needs. Formal market segmentation techniques, such as market studies and use cases, may be utilization at this stage. For inclusive design, the market is segmented based on the functional limitations of users. Moon and McAdams (2009) advocated the segmentation of a universal product family into groups of users with differing levels of impairment. Figure 19 shows the application of various platform leveraging strategies, developed by Meyer and Lehnerd (1997), to universal design.

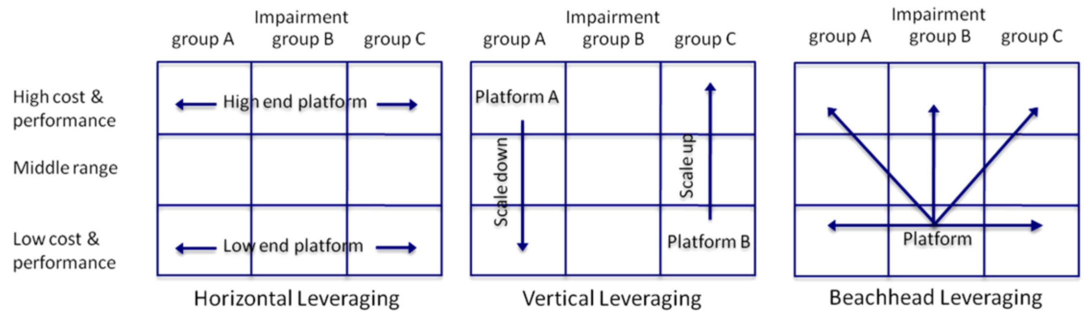


Figure 19: Platform leveraging strategies for universal design (Moon and McAdams 2009)

3.3.2 Action-Function Modeling

To identify limiting aspects of a design, we develop and analyze an action-function diagram for the typical product (Kostovich et al. 2009). An action-function diagram provides insight into the combination of product functionality and physical actions with which disabled users may find difficulty. In the design of a universal product family, the primary objective is to maximize the elements shared between products without harming product performance. Using an action-function diagram, allows the determination of which product features may stay the same and which need to be altered to address the needs of those with various impairments.

Changes to add accessibility to the action-function diagram can be identified either based on a designer's discretion or by utilizing the heuristics develop by Sangelkar et al. (2012). At this stage, designers may reference established universal design rules to ensure maximal inclusivity of the accessible design (Connell et al. 1997; University of Cambridge 2015). Possible changes to the action-function diagram include function changes, conceptual/morphological changes, or parametric changes. To ensure maximal commonality within the product family, differences between the action-function diagrams should be kept to a minimal.

3.3.3 Component Selection

Using the action-function diagrams, concepts are developed for the design of the products of the family. The goal of the concept development stage of the design is again to determine concepts that allow the maximum amount of sharing between products. Developing similar concepts for the product variants is integral to developing a product family with high commonality. However, if products are kept too similar, the performance of both product may be harmed. Concept selection for the product family requires the balance of concept commonality and expected performance.

After the concepts for the products are chosen, the products must be decomposed into the necessary design components. Again, the goal of this stage is to design the products in the family to use similar/the same components. Components can be established from the action-function diagram. Alternatively, system requirements, obtained from the needs of each market segment, may be utilized to select components. The House of Quality method (Hauser and Clausing 1988) is one such systematic technique for defining components to address system requirements.

Finally, the components which can be shared between products must be determined. Often, products will contain similar components, which ultimately cannot be shared due to differences in the parameters necessary to meet performance goals. Thus, this stage is a trade-off of product performance and commonality. Moon and McAdams (2012) provide a technique for product platforming based on usability and demand. Many similar methods exist for optimizing the balance of commonality and product performance.

3.3.4 Product Family Modularization

Finally, the product components are organized into modules and building block modules using the module clustering method presented in Section 2. In this approach, alternative modular architectures that possess both high commonality and optimal module boundaries are identified. From these alternatives, the modular design may be selected. This approach utilizes DSM clustering to evaluate modular architecture options along a Pareto front of maximum commonality and optimal modular architecture. The quality of the modular architecture is indicated using an index termed the modularity score. This approach allows for more ease in configuring and assembling product variants.

3.4 Case Study

To demonstrate the proposed design approach, a case study is presented of the design of a product family of typical and inclusive vehicle driver seats. The product family includes a variety of seats designed for use in different types of vehicles.

3.4.1 Market Segmentation

The market attack plan for our product family is to provide a high-end and low-end option that will be available in separate variants for those with and without moderate mobility impairment. The market segmentation plan is shown in Figure 20.

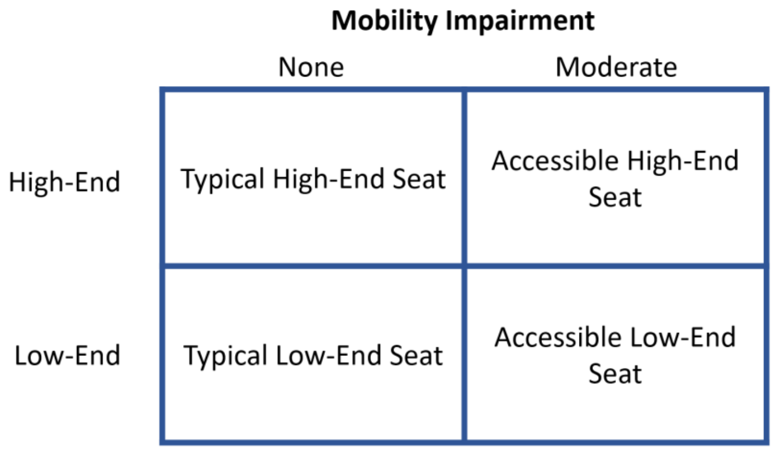


Figure 20: Targeted market segments for the driver seat case study

3.4.2 Action-Function Modeling

To identify design changes necessary to make a typical driver's seat more inclusive, an action-function diagram is created. The action-function diagram for the typical product is shown in Figure 21.

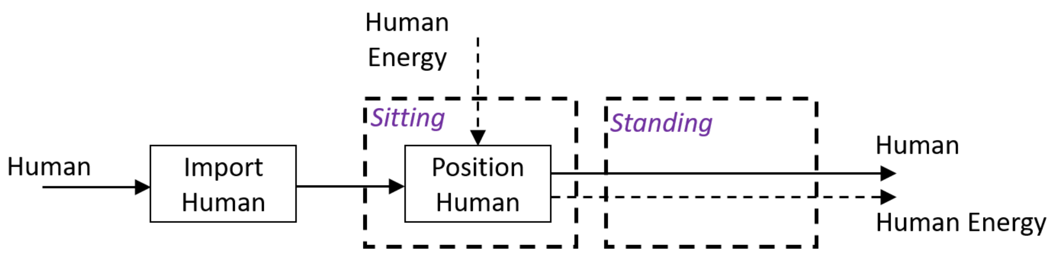


Figure 21: Action-function diagram of a typical vehicle driver seat

From the action-function diagram, we use the functional heuristics developed by Sangelkar et al. (2012) to identify potential changes to create an inclusive product variant.

The changes chosen for the inclusive product are presented in Figure 22. In this figure, diamond-shaped functions represent function additions in the inclusive design compared to the typical design.

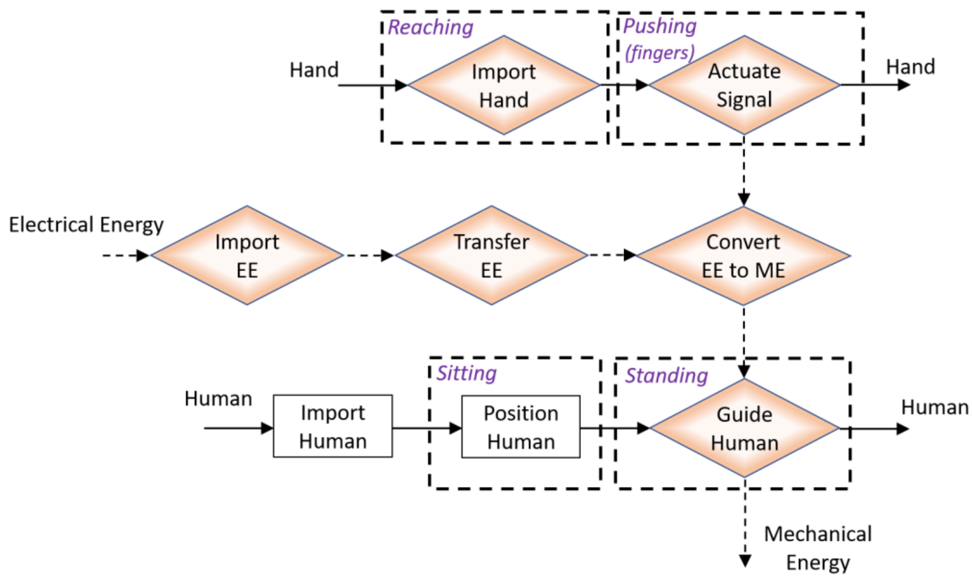


Figure 22: Action-function diagram of an inclusive vehicle driver seat

The alterations to the action-function diagram of the typical product include added functions that replace human energy with electrical energy to guide the human. Reducing the amount of human energy provides a much more accessible solution.

3.4.3 Component Selection

From the action-function diagrams, concepts are developed for the two products. The action-function diagram indicates that the components related to the seated position can be kept the same between the products. The typical product will be a standard driver seat with reclining and sliding functionality. The accessible product will contain the same standard seat components. However, the accessible seat will also possess a mechanism that allows it to both swivel and extend out of the vehicle through the use of electrical energy. These concepts possess similarities and similar components, yet provide the desired functions and performance. Figure 23 demonstrates the concept for the accessible products.

Figure 23: Design concept for the accessible products (Moon and McAdams 2012)

From these concepts, we develop a list of components. The list is designed to include a maximum number of common components. The set of components needed for the accessible product was developed, in part, using the work of Shi et al. (2009) as reference. The list of identified components needed to construct the product family is given in Table 7.

Table 7: Product list for the typical and inclusive vehicle driver seats

Component No.	Component Description	Component No.	Component Description
1	Upper Side Frame, L	31	Seat Cushion
2	Upper Side Frame, R	32	Seat Cushion Cover
3	Guide Stay, Head Rest, L	33	Head Cushion Rod, L
4	Guide Stay, Head Rest, R	34	Head Cushion Rod, R
5	Upper Back Frame	35	Seat Outer Finisher, L
6	Lower Back Frame	36	Seat Outer Finisher, R
7	Turning Rod	37	Sliding Assy, L
8	Rod Cover	38	Sliding Assy, R
9	Lower Side Bracket, L	39	Sliding Assy Handle
10	Lower Side Bracket, R	40	Seat Track
11	Front Seat Support	41	Extension Arm
12	Rear Seat Support	42	Housing Box
13	Reclining Lever	43	Lower Track, L
14	Spring, Reclining	44	Upper Track, L
15	Reclining Lever Stopper	45	Lower Track, R
16	Connecting Link, Reclining	46	Upper Track, R
17	Reclining Mechanism	47	Upper Arm Connection, L
18	Reclining Level Handle	48	Upper Arm Connection, R
19	Radial Spring, L	49	Extension Shaft Assy
20	Radial Spring, R	50	Base Plate
21	Lower R Spring Connection, L	51	Hook Plate
22	Upper R Spring Connection, L	52	Lunar Gear
23	Lower R Spring Connection, R	53	Foot Rest
24	Upper R Spring Connection, R	54	Hook Attachment
25	Lower Spring Support Assy	55	Premium Headrest Cushion
26	Upper Spring Support Assy	56	Premium Headrest Cover
27	Headrest Cushion	57	Premium Back Cushion
28	Headrest Cover	58	Premium Back Cushion Cover
29	Back Cushion	59	Premium Seat Cushion
30	Back Cushion Cover	60	Premium Seat Cushion Cover

The components needed to construct each product in the family are depicted in the schematics in Figure 24 and Figure 25. These figures show the general structure of the products identified prior to modularization.

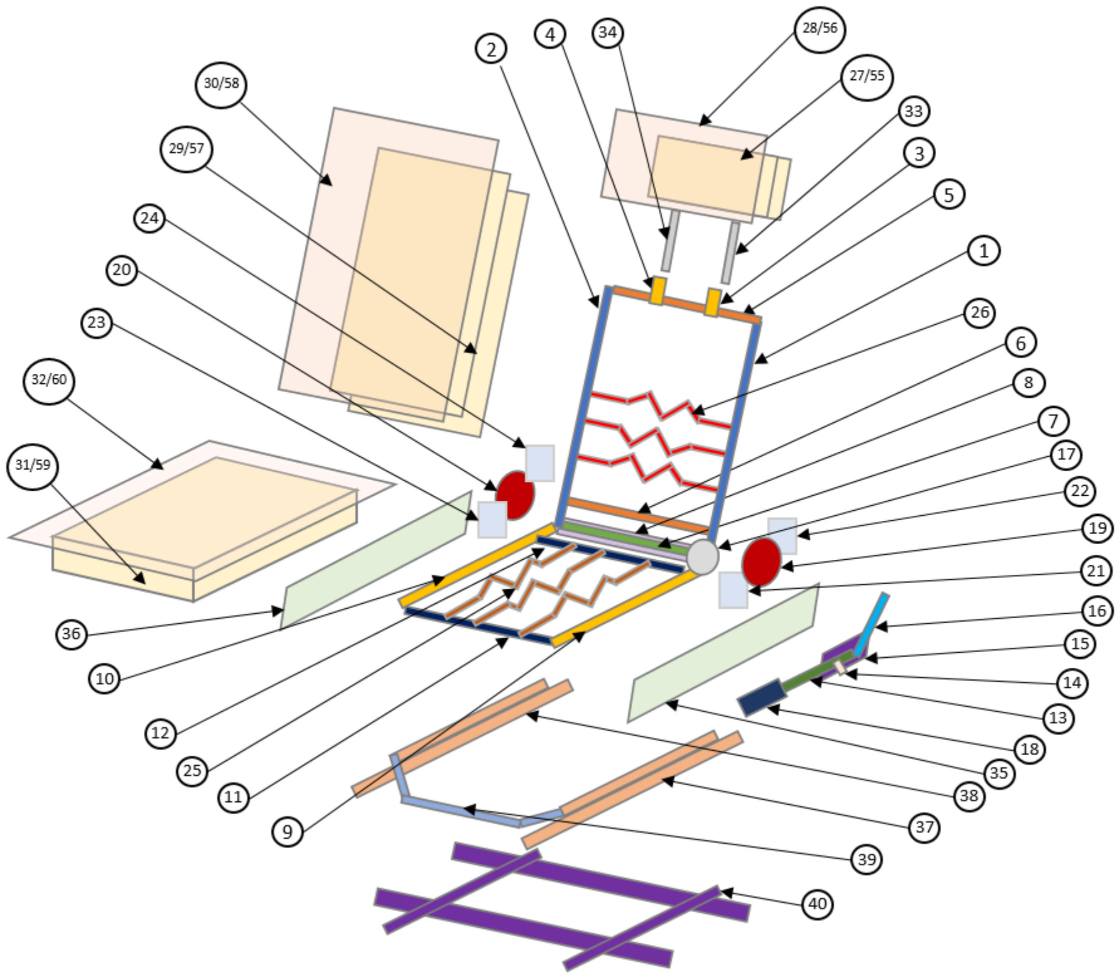


Figure 24: Components of the high- and low-end typical driver seats

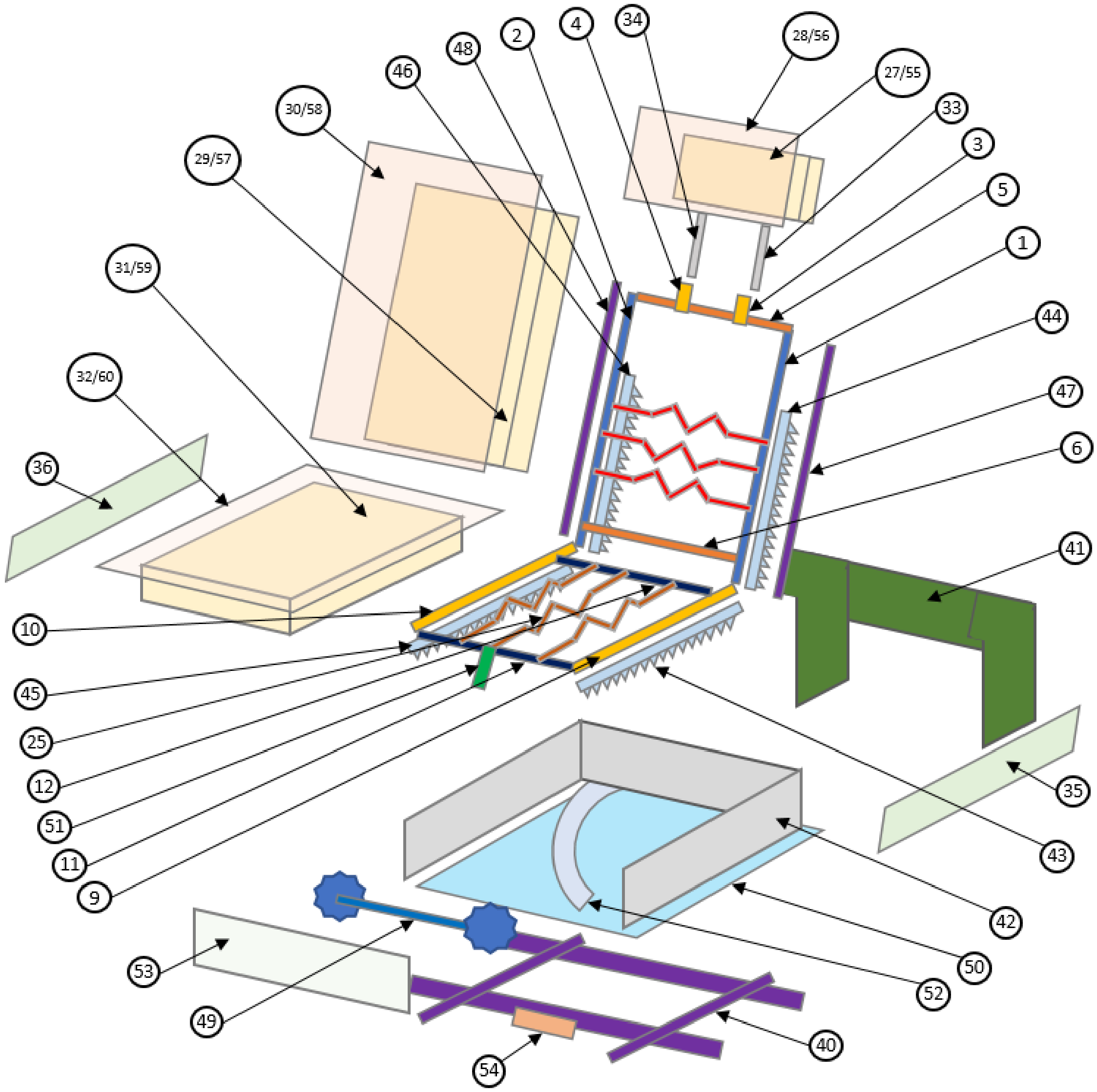


Figure 25: Components of the high- and low-end accessible driver seats

3.4.4 Product Family Modularization

To modularize the product family, a list of the components and their potential sharing between products is organized into a matrix. This product-component matrix (PCM) is included in Appendix B. A binary DSM is also constructed to represent interactions between components. The DSMs for each product in the family are combined into a single product family DSM. This DSM is shown in Figure 26.

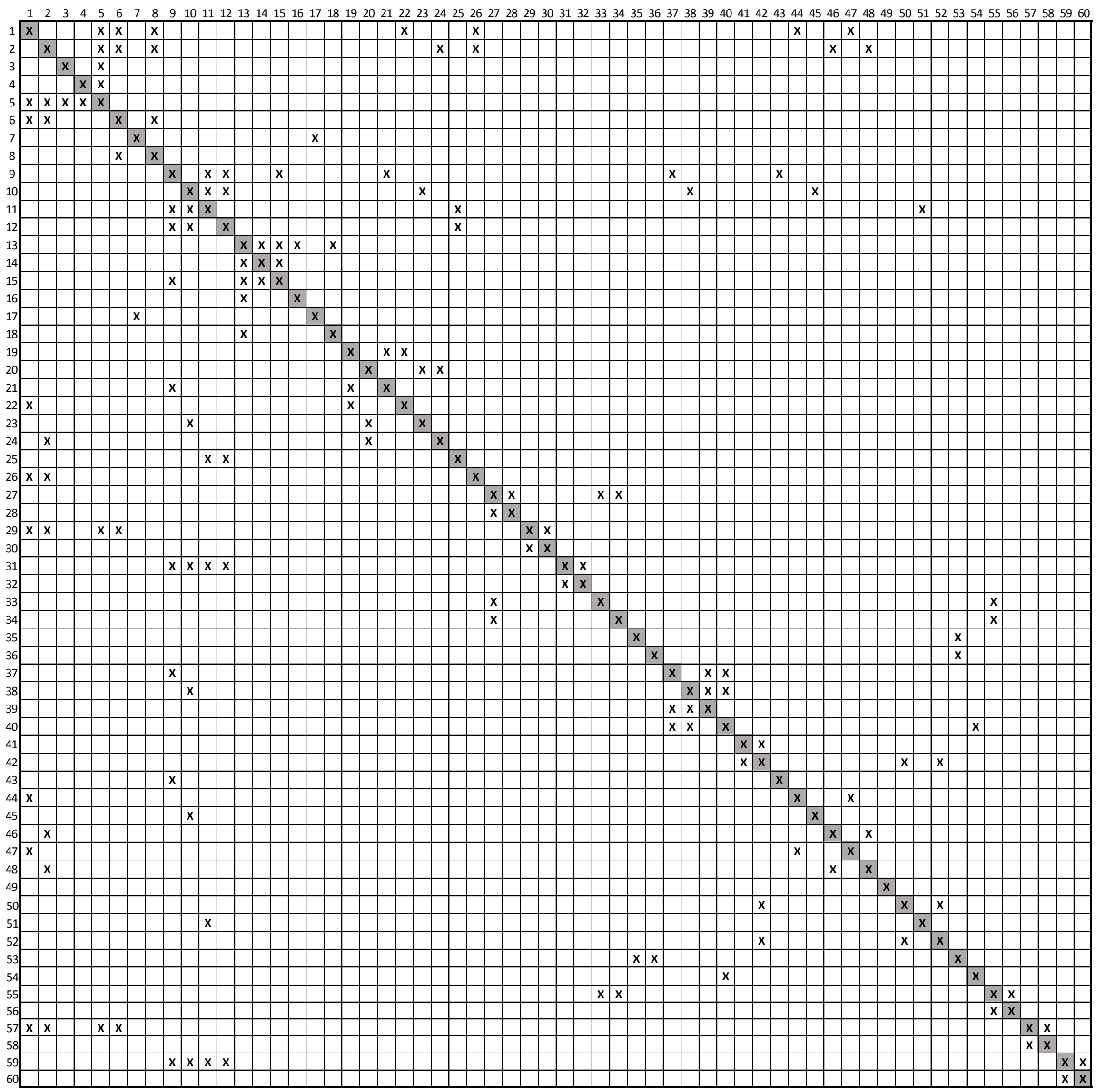


Figure 26: DSM for the product family of typical and accessible driver seats

The PCM and DSM for the product family were utilized to develop the Pareto front of alternatives for architecting the product family. To calculate the clustering cost for each modular architecture, the algorithm developed by Thebeau (2001) was used. To develop the front, the modularity score for each product was weighted equally in the calculation of the modularity score for the product family. The Total Constant Commonality Index (TCCI) was used to quantify commonality on a scale from 0, indicating no commonality, to 1, indicating high commonality. Figure 27 shows the front of architectures that provide maximum commonality and maximum modularity score.

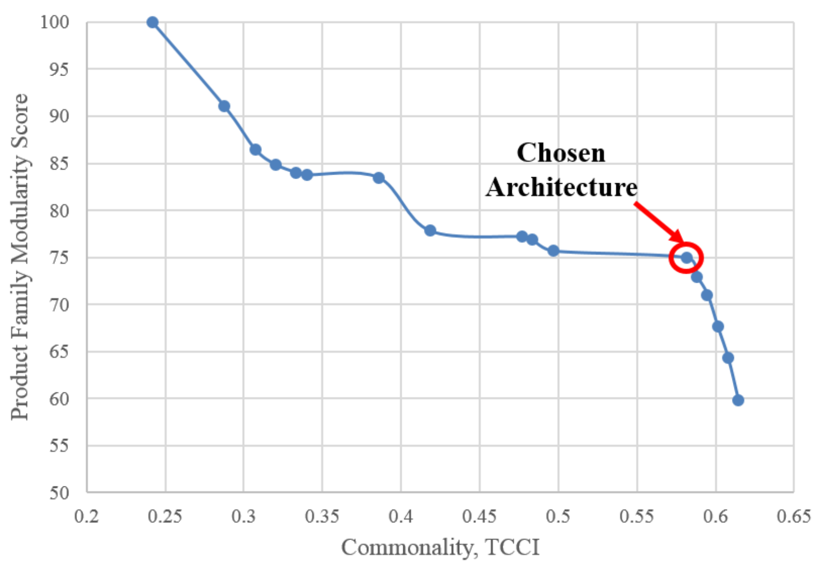


Figure 27: Pareto front of modular architectures with maximized commonality and optimal modular architecture for the driver seat case study

The architecture chosen for the design possesses a commonality of 0.58, which is relatively high with respect to the maximum possible commonality of 0.61. The design also has a weighted modularity score of 75.0. This design was chosen because of its high commonality. To obtain higher modularity scores, a large amount of commonality would need to be sacrificed past the chosen point. Thus, this appears to be a good option for prioritizing sharing within the product family.

The identified modular architecture consists of five groups of modules: universal, typical, accessible, high-end, and low-end modules. Universal modules are shared by all variants in the family. Typical, accessible, high-end, and low-end modules are shared only between the products that lie in each respective market categories. Table 8 provides the list of modules that constitute the product family.

Table 8: Product family modules

Module Type	Building Block No.	Module No.	Components
Universal	B1	M1	1 2 3 4 5 6 26
		M2	9 10 11 12 25
Typical	B3	M3	7 17
	B4	M4	13 14 15 16 18
	B5	M5	8
	B6	M6	19 21 22
	B7	M7	20 23 24
	B8	M8	35
	B9	M9	36
	B10	M10	37 38 39 40
Accessible	B11	M11	35 36 53
	B12	M12	40 54
	B13	M13	41 42 50 52
		M14	49
	B14	M15	43
	B15	M16	44 47
	B16	M17	45
	B17	M18	46 48
	B18	M19	51
Low-End	B19	M20	27 28 33 34
	B20	M21	29 30
	B21	M22	31 32
High-End	B22	M23	33 34 55 56
	B23	M24	57 58
	B24	M25	59 60

Using the modules identified in the chosen design, each of the products are modeled. Figure 28 displays models of the final modular design of each variant in the product family. Figure 29 through Figure 32 depict the construction of each of the identified modules.

(a)

(b)

(c)

(d)

Figure 28: Final modular design of the (a) typical low-end seat, (b) typical high-end seat, (c) accessible low-end seat, and (d) accessible high-end seat

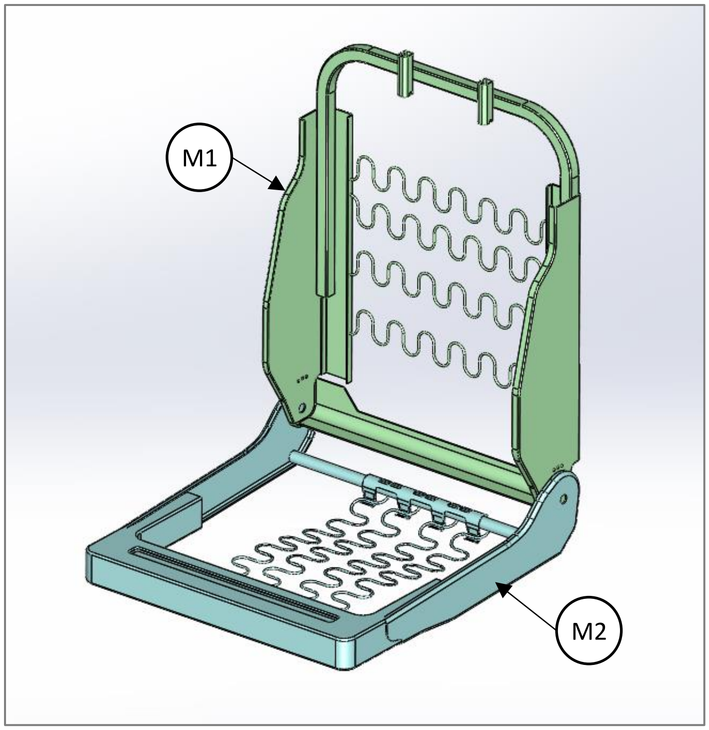


Figure 29: Universal modules

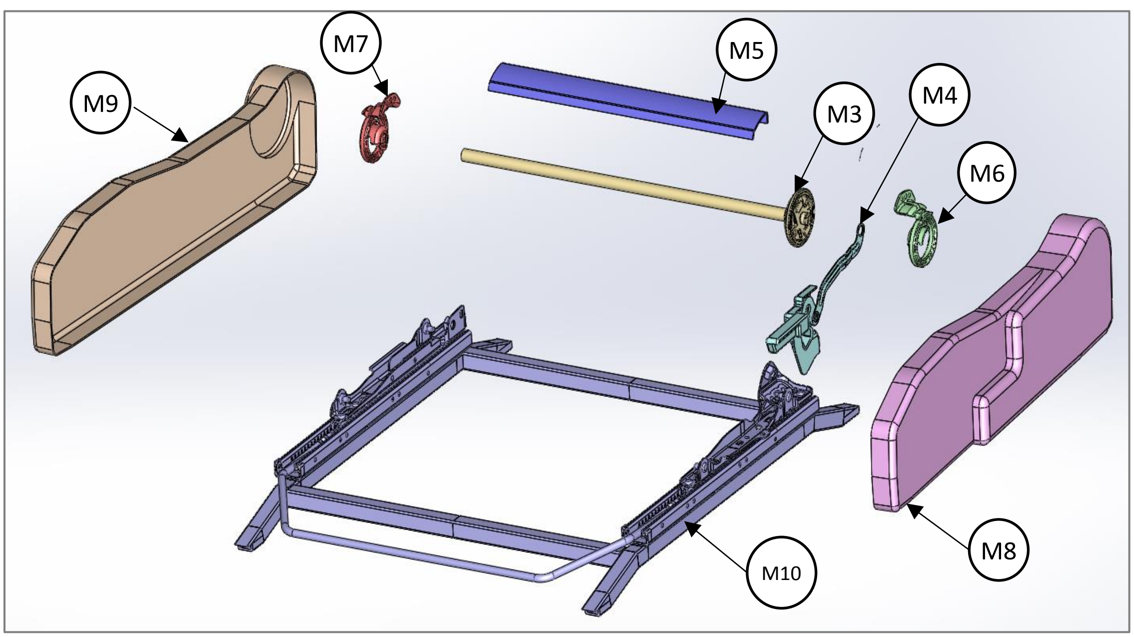


Figure 30: Typical modules

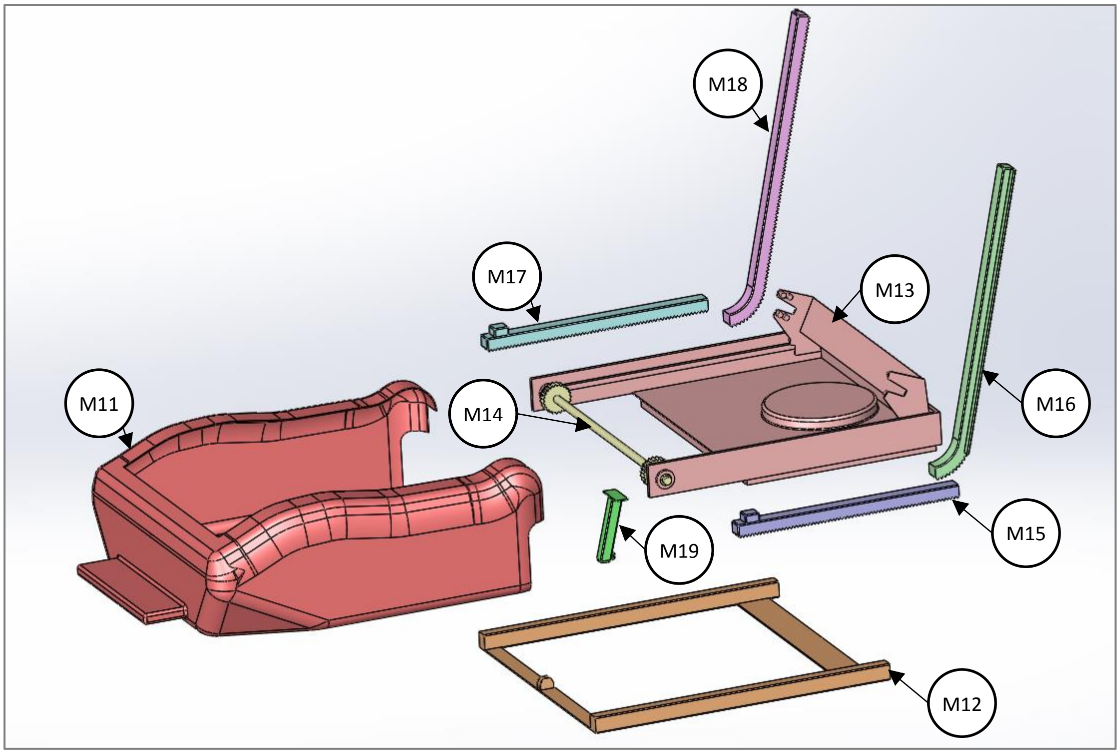


Figure 31: Accessible modules

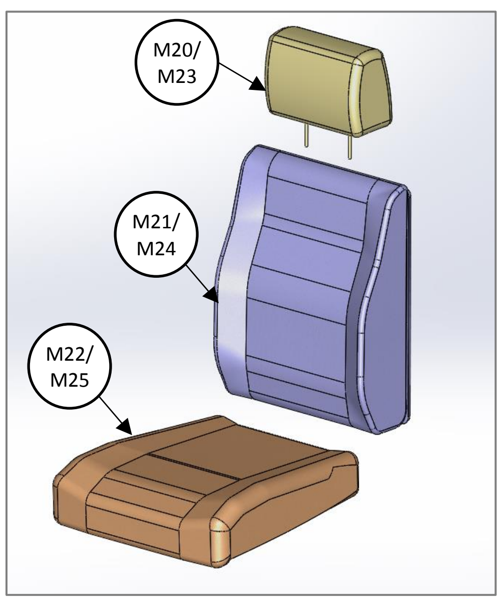


Figure 32: High- and low-end modules

3.5 Conclusions and Future Work

This article presents a strategy for the modular design and architecting of universal product families. The technique utilizes market segmentation, action-function modeling, and an algorithmic approach to generating modular architectures. The technique strategically analyzes potential modular architectures to determine a design that possesses the benefits of both commonality and modularity. The method aids in the development of product families that provide for typical and disabled users.

The method presented in this paper identifies modules that can be shared between various subsets of product variants in the family. This provides for a larger amount of sharing than typical market leveraging strategies. The approach provides benefits in targeting even more diverse markets. The proposed method could be further utilized to develop the modular architecture of products that target a third or higher dimension of the market, such as the inclusion of products for various vehicle models.

To improve upon the approach laid out in this paper, several advancements could be made. First, the ultimate quality of the design is highly dependent on the determination of product concepts that possess high commonality. Further research could be conducted into the most efficient changes to make to a typical product to maintain high commonality. The algorithmic clustering itself could also be improved. In the case study, a simple DSM that accounted for connectivity between components was utilized. Often, however, modules are desirable for strategic reasons other than just connectivity. Other clustering algorithms could be utilized in the analysis of modular architectures to account for additional concerns.

4 CONCLUSIONS

In this paper, two manuscripts were presented over the modularization of product families. The first manuscript presented the proposed method for architecting modular product families. This method utilized DSM clustering algorithms to evaluate and compare the quality of the modular architecture of various alternatives. Using this evaluation process and a part number based commonality index, design alternatives were identified along a Pareto front of maximum commonality and optimal modular architecture. From this front, proposed architectures can be selected and further refined. The use of this method was demonstrated in a case study of the design of a family of power tool. The second manuscript applied the proposed modular architecting method to use in the design of universal product families. In this paper, the component list needed for modularization was determined by first identifying market segments and action-function modeling. The universal product family design process was shown in a case study over the design of a typical and accessible vehicle driver seat.

REFERENCES

Bell, E. T. (1934). Exponential numbers. *The American Mathematical Monthly, 41*(7), 411-419. doi:10.2307/2300300

Borjesson, F., & Hölttä-Otto, K. (2014). A module generation algorithm for product architecture based on component interactions and strategic drivers. *Research in Engineering Design, 25*(1), 31-51. doi:10.1007/s00163-013-0164-2

Cameron, B. G., & Crawley, E. F. (2014). Crafting platform strategy based on anticipated benefits and costs, *Advances in Product Family and Product Platform Design* (pp. 49-70). New York, NY: Springer.

Connell, B. R., Jones, M., Mace, R., Mueller, J., Mullick, A., Ostroff, E., . . . Vanderheiden, G. (1997). The Principles of Universal Design, Version 2.0. Retrieved from NC State University, The Center for Universal Design website: https://www.ncsu.edu/ncsu/design/cud/about_ud/udprinciplestext.htm

Dahmus, J. B., Gonzalez-Zugasti, J. P., & Otto, K. N. (2001). Modular product architecture. *Design studies, 22*(5), 409-424.

Eppinger, S. D. (1997). *A planning method for integration of large-scale engineering systems.* Paper presented at the International Conference on Engineering Design.

Eppinger, S. D., & Browning, T. R. (2012). *Design structure matrix methods and applications.* Cambridge, MA: MIT Press.

Erixon, G. (1998). *Modular function deployment: a method for product modularisation*. Stockholm, Sweden: Royal Institute of Technology, Dept. of Manufacturing Systems, Assembly Systems Division.

Fisk, A. D., Rogers, W. A., Charness, N., Czaja, S. J., & Sharit, J. (2009). *Designing for older adults: Principles and creative human factors approaches*. Boca Raton, FL: CRC press.

Hauser, J. R., & Clausing, D. (1988). The house of quality. *Harvard Business Review, 66*(3), 63-73.

Helmer, R., Yassine, A., & Meier, C. (2010). Systematic module and interface definition using component design structure matrix. *Journal of Engineering Design, 21*(6), 647-675. doi:10.1080/09544820802563226

Höltä-Otto, K., Tang, V., & Otto, K. (2008). Analyzing module commonality for platform design using dendrograms. *Research in Engineering Design, 19*(2-3), 127-141. doi:10.1007/s00163-008-0044-3

Höltä, K. M., & Salonen, M. P. (2003). *Comparing three different modularity methods.* Paper presented at the ASME 2003 international design engineering technical conferences and computers and information in engineering conference.

Hsiao, S.-W., & Liu, E. (2005). A structural component-based approach for designing product family. *Computers in Industry, 56*(1), 13-28. doi:10.1016/j.compind.2004.10.002

Johnson, M. D., & Kirchain, R. (2010). Developing and assessing commonality metrics for product families: a process-based cost-modeling approach. *IEEE Transactions on Engineering Management, 57*(4), 634-648. doi:10.1109/TEM.2009.2034642

Kostovich, V., McAdams, D. A., & Moon, S. K. (2009). *Representing user activity and product function for universal design.* Paper presented at the ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference.

Lloyd-Sherlock, P. (2000). Population ageing in developed and developing regions: implications for health policy. *Social science & medicine, 51*(6), 887-895. doi:10.1016/S0277-9536(00)00068-X

Mace, R. (1985). Universal design: Barrier free environments for everyone. *Designers West, 33*(1), 147-152.

McAdams, D. A., & Kostovich, V. (2011). A framework and representation for universal product design. *International Journal of Design, 5*(1), 29-42.

Meyer, M. H., & Lehnerd, A. P. (1997). *The power of product platforms.* New York, NY: Simon and Schuster.

Moon, S. K., & McAdams, D. A. (2009). *A design method for developing a universal product family in a dynamic market environment.* Paper presented at the ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference.

Moon, S. K., & McAdams, D. A. (2012). A market-based design strategy for a universal product family. *Journal of Mechanical Design, 134*(11), 111007. doi:10.1115/1.4007395

Östgren, B. (1994). *Modularisation of the product gives effects in the entire production* (doctoral dissertation). The Royal Institute of Technology, Stockholm, Sweden.

Otto, K., Hölttä-Otto, K., & Simpson, T. W. (2013). *Linking 10 years of modular design research: alternative methods and tool chain sequences to support product platform design.* Paper presented at the ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference.

Robertson, D., & Ulrich, K. (1998). Planning for product platforms. *Sloan management review, 39*(4), 19-31.

Rojas Arciniegas, A. J., & Kim, H. M. (2011). Optimal component sharing in a product family by simultaneous consideration of minimum description length and impact metric. *Engineering Optimization, 43*(2), 175-192. doi:10.1080/0305215X.2010.486032

Sangelkar, S., Cowen, N., & McAdams, D. (2012). User activity–product function association based design rules for universal products. *Design studies, 33*(1), 85-110. doi:10.1016/j.destud.2011.06.002

Shi, Y., Lee, I.-T., Afsar, A. M., & Song, J.-I. (2009). Development and performance analysis of an automotive power seat for disabled persons. *International journal of automotive technology, 10*(4), 481-488. doi:10.1007/s12239-009-0055-8

Simpson, T. W., Siddique, Z., & Jiao, R. J. (2006). *Product platform and product family design: methods and applications*. New York, NY: Springer Science & Business Media.

Steward, D. V. (1981). The design structure system: A method for managing the design of complex systems. *IEEE Transactions on Engineering Management*(3), 71-74. doi:10.1109/TEM.1981.6448589

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

Story, M. F., Mueller, J. L., & Mace, R. L. (1998). *The universal design file: Designing for people of all ages and abilities.* Raleigh, NC: North Carolina State Univ., Raleigh. Center for Universal Design.

Sudjianto, A., & Otto, K. (2001). *Modularization to support multiple brand platforms.* Paper presented at the ASME International Design Engineering Technical Conferences, Design Theory and Methodology.

Thebeau, R. E. (2001). *Knowledge management of system interfaces and interactions for product development processes* (master's thesis). Massachusetts Institute of Technology, Cambridge, MA.

Thevenot, H. J., & Simpson, T. W. (2006). Commonality indices for product family design: a detailed comparison. *Journal of Engineering Design*, *17*(2), 99-119. doi:10.1080/09544820500275693

Ulrich, K. (1995). The role of product architecture in the manufacturing firm. *Research policy*, *24*(3), 419-440. doi:10.1016/0048-7333(94)00775-3

Ulrich, K., & Tung, K. (1991). *Fundamentals of product modularity*. Paper presented at the 1991 ASME Winter Annual Meeting Symposium on Issues in Design Manufacture/Integration. doi:10.1007/978-94-011-1390-8_12

United Nations, Department of Economic and Social Affairs, Population Division. (2015). *World Population Ageing 2015* (ST/ESA/SER.A/390). Retrieved from http://www.un.org/en/development/desa/population/publications/pdf/ageing/WPA2015_Report.pdf

Vanderheiden, G. (1997). Design for people with functional limitations due to disability, aging or circumstances In G. Salvendy (Ed.), *Handbook of Human Factors and Ergonomics, Second Edition*. New York, NY: John Wiley & Sons.

Vincent, G. K., & Velkoff, V. A. (2010). *The next four decades: The older population in the United States: 2010 to 2050.* Washington, DC: US Department of Commerce, Economics and Statistics Administration, US Census Bureau.

Yu, T.-L., Yassine, A. A., & Goldberg, D. E. (2007). An information theoretic method for developing modular architectures using genetic algorithms. *Research in Engineering Design*, *18*(2), 91-109. doi:10.1007/s00163-007-0030-1

Zamirowski, E. J., & Otto, K. N. (1999). *Identifying product family architecture modularity using function and variety heuristics.* Paper presented at the 11th International Conference on Design Theory and Methodology, ASME, Las Vegas.

APPENDIX A

Table 9 displays the PCM for the power tool case study referenced in Section 2.4. It contains the list of components that constitute each of the product variants in the study.

Table 9: PCM (transposed) for the case study over the design of a family of power tools

	Low-End Cordless Drill	Low-End Impact Driver	High-End Cordless Drill	High-End Impact Driver	
Clamshell	1	1	1	1	1
Armature 1	1	0	0	0	2
Armature 2	0	1	0	0	3
Stator Magnet 1	1	0	0	0	4
Stator Magnet 2	0	1	0	0	5
Motor Brushes	1	1	0	0	6
Brush Holders	1	1	0	0	7
Commutator	1	1	0	0	8
Rear Bearing	1	1	1	1	9
Pinion Gear	1	1	1	1	10
Front Bearing	1	1	1	1	11
Motor Fan	1	1	1	1	12
ESC	0	0	1	1	13
Magnetic Rotor 1	0	0	1	0	14
Magnetic Rotor 2	0	0	0	1	15
Stator 1	0	0	1	0	16
Stator 2	0	0	0	1	17
Trigger	1	1	1	1	18
Fwd/Rev Switch	1	1	1	1	19
VSR Switch LE	1	1	0	0	20
VSR Switch HE	0	0	1	1	21
Heat Sink	0	0	1	1	22
Electronics Board	0	0	1	1	23
Belt Clip	1	1	1	1	24
Bit Clip	1	0	1	0	25
Drill Light	0	0	1	0	26
Impact Driver Light	0	1	0	1	27
Chuck HE	0	0	1	0	28
Chuck LE	1	0	0	0	29
Transmission 1	1	0	0	0	30
Transmission 2	0	0	1	0	31
Impact Mech LE	0	1	0	0	32
Impact Mech HE	0	0	0	1	33
Anvil LE	0	1	0	0	34
Anvil HE	0	0	0	1	35
Nose Cone	0	1	0	1	36
Battery Connector	1	1	1	1	37
20V Battery	1	1	1	1	38
Grip LE	1	1	0	0	39
Grip HE	0	0	1	1	40
	1	2	3	4	

Figure 33 shows the optimal clustering of each product variant in the power tool case study referenced in Section 2.4. Each modular architecture alternative was compared with the clustering costs of these optimal clusterings to calculate the modularity scores of the products and product family.

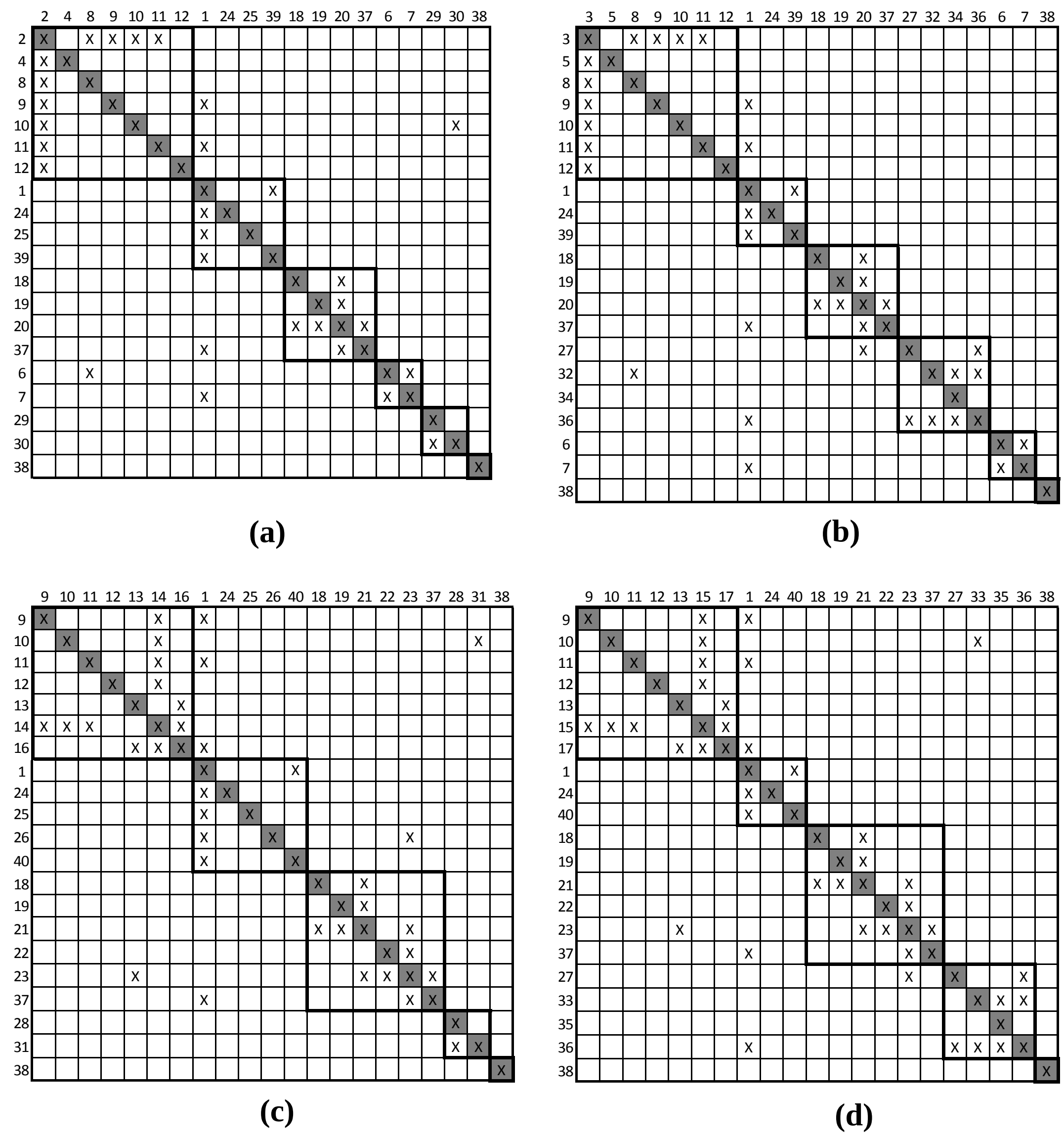


Figure 33: Optimally clustered DSM matrices for the (a) Low-End Cordless Drill, (b) Low-End Impact Driver, (c) High-End Cordless Drill, and (d) High-End Impact Driver

APPENDIX B

Table 10 displays the PCM for the vehicle driver seat case study referenced in Section 3.4.4. It contains the list of components that constitute each of the product variants in the study.

Table 10: PCM (transposed) for the case study over the design of typical and accessible vehicle driver seats

	Typical Seat, LE	Typical Seat, HE	Accessible Seat, LE	Accessible Seat, HE	
Side Frame, L	1	1	1	1	1
Side Frame, R	1	1	1	1	2
Guide Stay, Head Rest, L	1	1	1	1	3
Guide Stay, Head Rest, R	1	1	1	1	4
Top Frame	1	1	1	1	5
Lower Frame	1	1	1	1	6
Turning Rod	1	1	0	0	7
Rod Cover	1	1	0	0	8
Side Bracket, L	1	1	1	1	9
Side Bracket, R	1	1	1	1	10
Front Seat Support	1	1	1	1	11
Rear Seat Support	1	1	1	1	12
Reclining Lever	1	1	0	0	13
Spring, Reclining	1	1	0	0	14
Reclining Lever Stopper	1	1	0	0	15
Connecting Link, Reclining	1	1	0	0	16
Reclining Mechanism	1	1	0	0	17
Reclining Level Handle	1	1	0	0	18
Radial Spring, L	1	1	0	0	19
Radial Spring, R	1	1	0	0	20
Lower R Spring Connection, L	1	1	0	0	21
Upper R Spring Connection, L	1	1	0	0	22
Lower R Spring Connection, R	1	1	0	0	23
Upper R Spring Connection, R	1	1	0	0	24
Lower Spring Support Assy	1	1	1	1	25
Upper Spring Support Assy	1	1	1	1	26
Headrest Cushion	1	0	1	0	27
Headrest Cover	1	0	1	0	28
Back Cushion	1	0	1	0	29
Back Cushion Cover	1	0	1	0	30
Seat Cushion	1	0	1	0	31
Seat Cushion Cover	1	0	1	0	32
Head Cusion Rod, L	1	1	1	1	33
Head Cushion Rod, R	1	1	1	1	34
Seat Outer Finisher, L	1	1	1	1	35
Seat Outer Finisher, R	1	1	1	1	36
Sliding Assy, L	1	1	0	0	37
Sliding Assy, R	1	1	0	0	38
Sliding Assy Handle	1	1	0	0	39
Seat Track	1	1	1	1	40

Table 10: (Continued)

	Typical Seat, LE	Typical Seat, HE	Accessible Seat, LE	Accessible Seat, HE	
Extension Arm	0	0	1	1	41
Housing Box	0	0	1	1	42
Lower Track, L	0	0	1	1	43
Upper Track, L	0	0	1	1	44
Lower Track, R	0	0	1	1	45
Upper Track, R	0	0	1	1	46
Upper Arm Connection, L	0	0	1	1	47
Upper Arm Connection, R	0	0	1	1	48
Extension Shaft Assy	0	0	1	1	49
Base Plate	0	0	1	1	50
Hook Plate	0	0	1	1	51
Lunar Gear	0	0	1	1	52
Foot Rest	0	0	1	1	53
Hook Attachment	0	0	1	1	54
Premium Headrest Cushion	0	1	0	1	55
Premium Headrest Cover	0	1	0	1	56
Premium Back Cushion	0	1	0	1	57
Premium Back Cushion Cover	0	1	0	1	58
Premium Seat Cushion	0	1	0	1	59
Premium Seat Cushion Cover	0	1	0	1	60
	1	2	3	4	