A STRATEGIC MODULE-BASED PLATFORM DESIGN METHOD FOR
DEVELOPING CUSTOMIZED FAMILIES OF PRODUCTS AND SERVICES

A Dissertation in
Industrial Engineering
by
Seung Ki Moon

© 2008 Seung Ki Moon

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2008
The dissertation of Seung Ki Moon was reviewed and approved* by the following:

Timothy W. Simpson
Professor of Industrial and Mechanical Engineering
Thesis Adviser
Chair of Committee

Soundar R.T. Kumara
Allen, E., and Allen, M., Pearce Professor of Industrial Engineering
Co-Chair of Committee

David A. Nembhard
Associate Professor of Industrial Engineering

Dongwon Lee
Assistant Professor of College of Information Science Technology

Richard J. Koubek
Professor of Industrial Engineering
Head of the Harold and Inge Marcus Department of Industrial and Manufacturing Engineering

*Signatures are on file in the Graduate School
ABSTRACT

For mass customization environments, many companies are increasing their efforts to reduce cost and time for developing new products and services while satisfying individual customer needs. Most products these days include bundles of services to help satisfy customers’ needs and remain competitive in the market. Product family design facilitates mass customization by allowing highly differentiated products to be developed around a platform while targeting products to distinct market segments. Therefore, effective platforming of products and services is a cost-effective way to achieve mass customization. The objective in this research is to develop a Strategic Module-based Platform Design Method (SMPDM) to determine a platform design strategy to support product and service family design in a dynamic and uncertain environment. The proposed method extends concepts from product family and product platform design into service design, emphasizing a bottom-up approach and module-based design. Ontologies and object-oriented concepts are used to represent products and services and enable sharing and reuse of design information. Data mining techniques are used to identify a platform and modules by utilizing design information stored in a large database or repository. To determine a platform for family design in a dynamic and uncertain market environment, the SMPDM uses agent-based decision-making, involving a market-based negotiation mechanism and a game theoretic approach based on module-based platform concepts and a mathematical model. To demonstrate and validate the usefulness of the proposed method, it is applied to two industrial examples (a family of power tools and a family of checking account services) and tested in multiple scenario-based experiments. The
SMPDM provides an optimal platform design strategy that can be adapted to various dynamic and uncertain market environments. Therefore, the SMPDM can help develop design strategies to manage and create a cost-effective variety of products and services based on a platform in support of mass customization.
# TABLE OF CONTENTS

LIST OF FIGURES .............................................................................................................................................. vii

LIST OF TABLES .................................................................................................................................................. x

ACKNOWLEDGEMENTS ..................................................................................................................................... xii

Chapter 1  Introduction to Product and Service Family Design and Mass Customization ................................................................. 1

1.1 Families of Products and Services for Mass Customization .......... 1
1.2 Multi-Agent System and Data Mining for Product Development ...... 4
1.3 Research Focus in the Dissertation ................................................. 6
   1.3.1 Research Motivation ............................................................... 6
   1.3.2 Research Objectives ............................................................... 7
   1.3.3 Contributions from the Research .......................................... 8
1.4 Overview of the Dissertation .......................................................... 9

Chapter 2  Literature Review ......................................................................................................................... 14

2.1 Approaches to Product Family and Platform Design .................... 14
2.2 Designing Families of Products and Services ................................ 16
2.3 Design Knowledge and Data Mining ............................................ 18
2.4 Multi-Agent Systems and Their Applications ............................... 20
2.5 Game Theoretic Approaches ......................................................... 21
2.6 Limitations of Current Methods for Designing Families of Products and Services ............................................................. 23
2.7 Extensions to Service Design ....................................................... 25
2.8 Summary and Preview ................................................................. 28

Chapter 3  The Strategic Module-based Platform Design Method (SMPDM) .......................................................... 29

3.1 The Strategic Module-based Platform Design Method .................... 29
   3.1.1 Step 1 – Analyze Family Design Data .................................... 31
   3.1.2 Step 2 – Represent Design Knowledge .................................. 34
   3.1.3 Step 3 – Identify Modules and Platform Strategies ................. 35
   3.1.4 Step 4 – Specify Platform Design Strategy ............................ 36
3.2 Strategy for Verification and Testing of the SMPDM ....................... 39
3.3 Summary and Preview ................................................................. 42

Chapter 4  Design Knowledge Representation using an Ontology ............... 43

4.1 Overview of Ontologies in Product Knowledge Representation ........ 44
   4.1.1 Device ontology ........................................................................ 46
4.2 Techspecs Concept Ontology (TCO) .............................................................. 47
  4.2.1 Functional Level ................................................................................... 49
  4.2.2 Assembly Level .................................................................................. 51
  4.2.3 Product Technical Specifications ......................................................... 54
  4.2.4 Product Representation ..................................................................... 55
  4.2.5 Case Study: Family of Power Tools ..................................................... 56

4.3 Ontology for Service Design ........................................................................ 60
  4.3.1 Service Analysis ................................................................................... 61
  4.3.2 Service Process Model ......................................................................... 65
  4.3.3 Service Representation ....................................................................... 67
  4.3.4 Case Study: Family of Checking Account Services ............................. 69

4.4 Summary and Preview .................................................................................. 73

Chapter 5  Module and Platform Identification using Data Mining .................. 74
  5.1 A Method for Module and Platform Identification ..................................... 75
    5.1.1 Phase 1: Association Rule Mining for Knowledge Acquisition ........... 76
    5.1.2 Phase 2: Module Determination ........................................................... 78
    5.1.3 Phase 3: Module and Platform Identification ....................................... 81
  5.2 Case Study I: Power Tool Family ............................................................... 83
    5.2.1 Phase 1: Association Rule Mining for Knowledge Acquisition .......... 85
    5.2.2 Phase 2: Fuzzy Clustering for Defining Modules .............................. 87
    5.2.3 Phase 3: Module and Platform Identification ....................................... 90
  5.3 Case Study II: Checking Account Services ................................................ 92
    5.3.1 Phase 1: Knowledge Representation .................................................... 92
    5.3.2 Phase 2: Fuzzy Clustering for Defining Modules .............................. 95
    5.3.3 Phase 3: Modules and Platform Identification ..................................... 96
  5.4 Summary and Preview ................................................................................ 98

Chapter 6  Agent-based Decision-Making for Determining a Platform Strategy .... 99
  6.1 Module-based Platform and Mathematical Model for a Product Family ...... 100
    6.1.1 Engineering Parameter Cost Model for Product Family .................... 100
    6.1.2 Mathematical Model for Platform Level Selection based on EP Cost .... 103
  6.2 Dynamic Multi-Agent System ................................................................... 107
    6.2.1 Electronic Market ................................................................................. 108
    6.2.2 System Architecture ............................................................................. 109
    6.2.3 Task Decomposition ............................................................................ 111
  6.3 Negotiation based on Market Mechanisms ............................................... 113
    6.3.1 Negotiation Protocol between PFA and MDAs .................................. 116
    6.3.2 Negotiation Mechanism between MDAs and CPAs ........................... 117
    6.3.3 Case Study: Power Tool Family ........................................................... 119
  6.4 Game Theoretic Approach for Strategic Platform Design ......................... 127
    6.4.1 A Bayesian Game with Incomplete Information ............................... 128
LIST OF FIGURES

Figure 1.1: Classification based on Good’s and Companies ...........................................3
Figure 1.2: Pictorial Overview of the Dissertation ..........................................................11
Figure 2.1: Literature Review and Research Domain ......................................................24
Figure 2.2: Process of Service Family Design and Service Families .............................28
Figure 3.1: Steps and Tools of the SMPDM .................................................................30
Figure 3.2: The SMPDM in a Multi-Agent Decision-Making Framework ....................31
Figure 3.3: The Process of Developing a Family of Products and Services ..................32
Figure 3.4: Functional Model of a Circular Saw and Five Modules ..............................33
Figure 3.5: Functional Hierarchy and Product Representation using an Ontology .........34
Figure 3.6: Multi-Agent System Architecture based on a Dynamic Market Environment .................................................................37
Figure 4.1: Roadmap for Chapter 4 .............................................................................44
Figure 4.2: Overall Functions of Main Components and Device Ontology .................47
Figure 4.3: Hierarchy of Techspecs Concept Ontology ..................................................48
Figure 4.4: Functional Model of a Circular Saw ...........................................................50
Figure 4.5: Functional Hierarchy of a Circular Saw and Properties ..............................51
Figure 4.6: Physical Decomposition Hierarchy of Circular Saw and Property .............52
Figure 4.7: Connection Information between Devices based on Conduit ..................52
Figure 4.8: Overall Relation between Hierarchies .......................................................53
Figure 4.9: Functional Hierarchy and Product Representation using TCO ..................56
Figure 4.10: The Delta Power Tool Family .................................................................57
Figure 4.11: The Delta Power Tool Classes and Its Subclasses of ‘Function’ ...............58
Figure 4.12: Examples of Rules and Facts for Design Knowledge ..............................59
Figure 4.13: The Process of Developing Service Ontology ........................................60
Figure 4.14: Saving Service Process and Applied to Object-Oriented Concepts .......62
Figure 4.15: Extension Object-Oriented Concepts to Service Design .......................64
Figure 4.16: Example for Inheritance in Service Analysis and Design .....................65
Figure 4.17: An Example of a Sequence Diagram for a Banking Service ...............66
Figure 4.18: Service Process Model and Properties for a Node ..............................67
Figure 4.19: Hierarchy of the Service Ontology .......................................................69
Figure 4.20: Activity Diagram for a Deposit Process Function ..............................71
Figure 4.21: Service Process Model for a Deposit Service Function ......................72
Figure 4.22: Checking Account Service Classes and Subclasses ..........................72
Figure 5.1: Road Map for Chapter 5 .......................................................................74
Figure 5.2: Method for Platform and Module Identification .................................76
Figure 5.3: Process of Classifying Clusters ..............................................................81
Figure 5.4: Fuzzy Membership Function Representing Platform Level ...................82
Figure 5.5: Process of Association Rule Mining .....................................................86
Figure 5.6: Input / Output of Magnum Opus for Power Tool Family ....................86
Figure 5.7: Values of PC for Fuzziness .................................................................87
Figure 5.8: Values of the PC for Three Different Initial Seeds ..............................88
Figure 5.9: Examples of Integration Rule and Split Rule ......................................89
Figure 5.10: Values of the PC for Three Different Initial Seeds ............................95
Figure 6.1: Road Map for Chapter 6 ......................................................................100
Figure 6.2: Trade-off in Platform Level Selection in terms of EP costs .................103
Figure 6.3: Electronic Market Environment ..........................................................109
Figure 6.4: Proposed Dynamic Multi-Agent System for Product Family Design ......110
Figure 6.5: The Roles of CTA and CDA for Task Decomposition ............................. 112
Figure 6.6: Relationships among the Agents Where Designing Modules ................... 113
Figure 6.7: Negotiation Process for Two MDAs: An Example.................................. 116
Figure 6.8: The Proposed Bayesian Game for Strategic Platform Design ................. 133
Figure 6.9: The Payoffs of Agent 1 when Customers’ Preference is 0.6..................... 138
Figure 6.10: The Payoffs of Agent 2 when Customers’ Preference is 0.6............... 138
Figure 6.11: Payoffs of Agent 1 when Customer’s Preference is 0.4......................... 144
Figure 6.12: Payoffs of Agent 1 when Customer’s Preference is 0.5....................... 145
LIST OF TABLES

Table 2.1: Examples of Module-based and Scale-based Product Families .......... 16
Table 2.2: Possible Methods for Service Family Design and Limitation ............... 27
Table 3.1: Agents’ Roles and Knowledge for the Proposed Multi-Agent System ..... 38
Table 3.2: Decision-Making Methods for Agents ................................................. 39
Table 3.3: Relationship between the SMPDM and Chapters ................................. 40
Table 4.1: Properties of Technical Specifications .................................................. 54
Table 4.2: A Function-Process Matrix for Service Analysis ................................. 61
Table 4.3: Object-Oriented Concepts and Applications for Service Design .......... 63
Table 4.4: Four Checking Account Services in a Banking Service Family .......... 70
Table 4.5: The Function-Process Matrix for Four Checking Account Services .... 70
Table 5.1: Product Representation for Power Tool Family ................................. 84
Table 5.2: FCM Results (Membership Value) for 13 Clusters .............................. 88
Table 5.3: New Platform and Modules for Five Products (Platform Level Value) .... 91
Table 5.4: Comparison of Existing Products and Proposed Products .................... 91
Table 5.5: Property Codes for Process Features in the Four Checking Account Services .......................................................... 93
Table 5.6: Service Representation for Four Checking Account Services ............... 94
Table 5.7: Clustering Results for the Four Checking Account Services ............... 96
Table 5.8: New Platform and Modules for the Family of Checking Accounts ...... 97
Table 6.1: EP Costs of an EP Module for Product Family Design ....................... 102
Table 6.2: Risk Value Analysis between the Utility Value of $T_1$ and $T_2$ (Risk Value) .......................................................... 117
Table 6.3: The Results of Analyzing CNs and FRs for the Power Tool Family ...... 121
Table 6.4: The Results of Tasks Decomposition .................................................. 122
Table 6.5: Initialization of MDAs for a Circular Saw ......................................................123
Table 6.6: The Result of the First Bidding with CPAs for EP values ...............................124
Table 6.7: Deals for a Negotiation with PFA ................................................................124
Table 6.8: Set of Deals for the Platform ......................................................................125
Table 6.9: The Constraints and Cost of Subtask 2 for Products .................................126
Table 6.10: The Result of Negotiation between CDA and STA2, 3 .............................127
Table 6.11: Strategies for EP Module Design ..............................................................135
Table 6.12: Expected Strategy Costs and Market Prices (MSR and CP =1) ...............136
Table 6.13: A Bayesian Game for Determining a Design Strategy ..............................137
Table 6.14: Platform Strategies Based on Service Process Modules ..........................142
Table 6.15: Expected Strategy Cost and Market Price (MSR and CP = 1) .................143
Table 6.16: A Bayesian Game for Determining a Service Platform ...........................143
Table 7.1: Customer Needs and Functional Requirements for Circular Saw ..........148
Table 7.2: Product Representation for Circular Saw ....................................................149
Table 7.3: Module Categorization and Design Strategies for the Power Tool Family .................................................................150
Table 7.4: EP Modules’ Components and Design Strategies ....................................151
Table 7.5: Platform Design Strategy for EP Modules ..................................................151
Table 7.6: Service Process Representation for Plan A ...............................................153
Table 7.7: Module Categorization and Design Strategies for the Checking Account Services ......................................................................................................154
Table 7.8: Design Strategies for the Checking Account Service Family ..................155
ACKNOWLEDGEMENTS

I would like to thank my family for everything that they've done for me all this time. Especially, my wife, Eun Mi, was always with me and believed in me through the Ph.D. study at Penn State. I honestly could not have done any of this without her. And, my son, Jong Hyun, always bring pleasure to me during the study. I am greatly indebted to my parents and parents-in-laws who supported my family and encouraged me to achieve higher goals throughout my work. I would like to share this achievement with them.

I would like to express my gratitude to Dr. Timothy W. Simpson, Professor of Industrial Engineering and Mechanical Engineering, for his advice, insights, support, and encouragement throughout my Ph.D. study. He provided me with many opportunities to enhance my knowledge in academic area through numerous conferences. I learned a lot from him in these past years. I do not know how I would have done my work without you. I am also grateful to Dr. Soundar R. T. Kumara, Allen, E., and Allen, M., Pearce Professor of Industrial Engineering, who was to advise a direction of my research, to give me the confidence in my research, and to support me with many opportunities toward innovation. I would also like to thank Dr. Jun Shu, Assistant Professor of Supply Chain and Information System, the SMEAL College of Business, for providing guidance and feedback on research problems and issues studied in my research. I would also like to thank the other committee members, Dr. David A. Nembhard, Associate Professor of Industrial Engineering, and Dr. Dongwon Lee, Assistant Professor of College of
Information Science Technology, for their guidance and suggestions during my Ph.D. study.

I would like to thank current students and alumni here in IE of Penn State. I also owe special thanks to Korean IE students who spent a lot of time at State College and talk with me about research, life, and future. I really can't thank you enough for all their help. I would like to thank all EDOG and LISQ members for supporting the work and sharing memories with me in my life.

Finally, I would like to thank Penn State for providing me the opportunity to pursue education and conduct my research. I would also like to gratefully acknowledge support from the National Science Foundation, Grant No. IIS-0325402 (ITR Program).
Chapter 1

Introduction to Product and Service Family Design and Mass Customization

The principal objective in this dissertation is to develop the Strategic Module-based Platform Design Method (SMPDM) to determine a platform design strategy to support product and service family design in a dynamic and uncertain market environment. This chapter provides the research motivation and foundation for developing this method by focusing on mass customization of products and services. In specific, the concepts of product family and product platform design are extended to service family design in Section 1.1. Multi-agent systems and data mining related product development are presented in Section 1.2. In Section 1.3, the research objectives and contributions of the work are described. Finally, Section 1.4 provides an overview of the dissertation.

1.1 Families of Products and Services for Mass Customization

Mass customization depends on a company’s ability to provide customized products or services based on economical and flexible development and production systems [78]. For mass customization, companies are increasing their efforts to reduce cost and lead-time when developing new products and services while satisfying individual customer needs. By sharing and reusing assets such as components, processes, information, and knowledge across a family of products and services, companies can
efficiently develop a set of differentiated economic offerings by improving flexibility and responsiveness of product and service development [79]. Product family design is a cost-effective way to achieve mass customization by allowing highly differentiated products to be developed from a common platform while targeting products to distinct market segments [76].

Recent trends seek to apply and extend principles from product family design into new service development [78]. Families of services and service platforms have been developed and applied in various service industries based on design theories and methodologies for products and manufacturing systems [32, 52]. For example, a bank can develop a family of checking account services by combining common services (e.g., Deposit, Withdraw, and Banking Statement), and various options (e.g., Trade Stocks Online, Maintenance Fee, and Loans) for offering different services. The common services are considered as a service platform for the service families. A typical approach to create a variety of services is to provide customers with various options and choices related to individual customer needs, which often warrant additional charges as they add value to the initial offering [39]. For example, in the IBM Malaysia service unit, the modularization of the scope of work and processes has been applied to service level design for mass customization [65]. Meanwhile, Lincoln Re used platform concepts to develop new insurance services [52].

As shown on Figure 1.1, most products in industries increasingly include additional services to satisfy customers’ needs and survive in today’s competitive market
environment. For example, GM introduced its OnStar\(^1\) service to provide customized maintenance and safety services. Apple-iPod+iTunes\(^2\) provides various customized services including operating software and accessories for its families of iPod MP3 players. Apple and Nike\(^3\) develop new products for customized exercise services by combining their products. Therefore, services should be considered as an important factor during product design. These paradigms in products and services have shifted in many industries that produce customized goods and are interested in maximizing resource utilization by sharing and reusing distributed design knowledge and information when developing new goods.

---

\(^1\) [http://www.onstar.com](http://www.onstar.com)


\(^3\) [http://www.nike.com](http://www.nike.com)
1.2 Multi-Agent System and Data Mining for Product Development

Electronic markets and web-based supply chain management have improved traditional product development processes by increasing the participation of customers and applying various trading processes between companies and suppliers in dynamic market environments. With the economic potential of reducing transaction costs, the applications of electronic markets are dramatically increasing in many industries [7]. The growing number of electronic markets for product development has significantly increased the amount of information related to the design and complexity of transactions, making it difficult to control the electronic markets with human resources [63]. In recent years, agents and multi-agent systems (MASs) have become a powerful and prevalent methodology for investigating and developing complex systems to support system designers and analyzers as discussed next.

Multi-agent systems have become increasingly widespread and accepted as one of the ideal mechanisms to efficiently develop various products through integrating distributed design knowledge and information [11]. An agent has access to at least one and potentially many sources of information and is able to collate and manipulate information obtained from these sources in order to answer queries posed by users and other information agents [94]. Since information integration can be facilitated through task decomposition, collaboration, and negotiation during product design [75], agent-based technologies provide a natural and useful means to achieve information integration in a distributed environment [46]. An agent-based technique based on agents’ roles and tasks can provide appropriate methods to solve product design problems [11, 67, 94].
Agents have been used extensively in product design and can be used in product family design if developed properly.

Data mining has been defined as the process of extracting valid, previously unknown, and easily interpretable information from large databases in order to improve and optimize engineering design and manufacturing process decisions [12, 19]. In mass customization, data mining can be used to help identify customer needs, to find relationships between customer needs and functional requirements, and to cluster products based on functional similarity to facilitate modular design [12]. At the conceptual design stage, data mining can aid decision-making when selecting design concepts by extracting design knowledge and rules, clustering design cases, and exploring interactively conceptual designs in large product development databases [12].

For global sourcing and manufacturing, product development is highly dependent on knowledge-intensive and collaborative systems for building on specialized knowledge across nations, organizations, and professions to develop customized products for different market segments [42, 87]. Knowledge-intensive and collaborative support has been increasingly important in product development to maintain future competitive advantage [100]. Knowledge support systems can provide a solution for iterative design and manufacturing activities that are performed by sharing and reusing knowledge related to product development. In knowledge support and management systems, data mining approaches can facilitate extraction of information in design repositories to generate new knowledge for product development.
1.3 Research Focus in the Dissertation

The research motivation introduced next provides research objectives and outlines contributions that embody the deliverables from the research. The research objectives discussed in Section 1.3.2 are used to structure the research and define the verification methods for this work. The resulting research contributions are introduced in Section 1.3.3.

1.3.1 Research Motivation

Innovative companies that generate a variety of products and services for satisfying customers’ specific needs are invoking and increasing research on customized services, but the majority of their efforts are still focused on products and manufacturing operations [78]. Recently, theories and methodologies for customized products are being applied to service development [32], and the concept of product family design, in particular, provides feasible solutions in various customized service industries [32, 52, 65]. The value of products and services depends on market segmentation strategies that are identified by information derived from the relationship between customer needs and providers [25]. In dynamic and uncertain market environments, however, we only have incomplete or uncertain information regarding market trends, customer’s preferences, production costs, and a company’s strategies for product development. To facilitate customized product and service design in dynamic and uncertain market environments, we consider strategic module sharing for designing a platform in a product and service family. The present work is motivated by the need to develop platform design strategies
for customized families of products and services in dynamic and uncertain market environments. The platform design strategies can be defined as alternative functional combinations for providing a variety of products or services based on market trends, customer’s preferences, production costs, and a company’s strategies. For service family design, the proposed method is developed based on extending concepts from product platform design to service design. The motivation for this research is elaborated more in the literature review in the next chapter.

1.3.2 Research Objectives

The principle goal in this dissertation is to develop a method to facilitate the design of a module-based platform for customized families of products and services in a dynamic and uncertain market environment. Data mining and agent-based decision-making – particularly negotiation and a game theoretic approach – provide the foundation on which this method is built. Keeping the primary research objective as the focus, the following secondary objectives are investigated:

- Extend concepts from platform-based product design to develop a module-based service family.
- Develop ontologies for products and services to represent semantic meaning for their functional features for family design.
- Identify a platform along with variant and unique modules using data mining techniques.
• Develop a multi-agent system to support platform design for product and service families in a dynamic market environment.

• Apply agent-based decision-making to generate and determine platform design strategies.

The secondary objectives are stated here primarily to provide context for the literature review in the next chapter and the development of the Strategic Module-based Platform Design Method (SMPDM) in Chapter 3.

1.3.3 Contributions from the Research

The main contribution from this research is the development of the Strategic Module-based Platform Design Method (SMPDM) for determining a platform to support product and service family design in dynamic and uncertain market environments. The SMPDM employs agent-based decision-making using market mechanisms to improve individual customer’s satisfaction through a variety of products and services in a dynamic market environment. Some of the major contributions include:

Platform design for families of products and services - most research has focused on designing a product platform. In this research, concepts from product platform design are extended to service platform design to support a family of services.
**Decision-making based on market mechanisms** - Market mechanisms in decision-making are applied to platform design to reflect dynamic design factors and reduce computational efforts. The proposed market-based negotiation mechanism can be applied to a web-based design system for recommending design information in a distributed design environment. For uncertain market environments, a game theoretic approach is used to determine an optimal platform design strategy.

**Agent-based methodology** - Unlike most of other reported research, the proposed agent-based system has dynamic agents that depend upon the product’s characteristics and designer’s strategies. The dynamic agent system can be applied to various areas for developing and improving a family of products and services.

**Knowledge representation and discovery for design** - For information related to products and services, an ontology and data mining techniques are used to represent and discover design knowledge. Specifically, object-oriented concepts are applied to analyze and represent services. The proposed method can be applied to develop design knowledge in a large database or repository and agent’s knowledge for reasoning and planning in an agent-based system.

**1.4 Overview of the Dissertation**

An overview of the chapters in the dissertation is shown in Figure 1.2. It proceeds from bottom to top, beginning with the foundation provided in Chapters 1 and 2: Module-
Based Platform Design and Agent-Based Decision Making. This figure provides a roadmap for the dissertation to help guide the reader through the work as the research progresses from chapter to chapter. Having laid the foundation by introducing the research motivation and objectives for the work in this chapter, the next chapter contains a literature review of related research, elucidating the problems and opportunities in family and platform design of products and services. Research areas related to this work include: (1) strategies and methods for product family and platform design (see Section 2.1), (2) applications for family design in products and services (see Section 2.2), (3) design knowledge and data mining (see Section 2.3), (4) multi-agent systems for product design (see Section 2.4), and (5) game theoretic approaches for uncertain market environments (see Section 2.5). Current limitations in designing families of products and services are described in Section 2.6. Based on the literature review, a discussion of how to extend the concepts from product family design methods to service family design is offered in Section 2.7.

The SMPDM is introduced in Chapter 3 as elements from Chapters 1 and 2 are synthesized into a method for determining a platform for customized families of products and services in dynamic and uncertain market environments. The SMPDM and its associated steps are introduced in Section 3.1. Section 3.2 presents an outline of the strategy for verification and testing of the proposed method.
To share and reuse the information related to design, ontologies are investigated to represent design knowledge for products and services in Chapter 4. To represent
products and components, Techspecs Concept Ontology (TCO) is employed and demonstrated in Section 4.2. For services, an ontology and object-oriented concepts are used to represent design knowledge as described in Section 4.3.

In Chapter 5, data mining techniques are applied to develop a method for identifying a platform along with variant and unique modules in a family of products and services. Section 5.1 introduces the foundations of data mining techniques related to the proposed method, namely, clustering, association rule mining, and classification. The proposed method is presented and demonstrated using a power tool family in Section 5.2. For services, a family of checking account services is used to demonstrate the proposed method in Section 5.3.

To determine a platform for family design in dynamic and uncertain market environments, various agent-based decision making approaches are investigated in Chapter 6. Module-based platform design and a mathematical model are introduced in Section 6.1. The architecture of a dynamic multi-agent system (DMAS) is proposed along with specific agents’ roles and knowledge in Section 6.2. A market-based negotiation mechanism is introduced to determine a platform in a dynamic market environment in Section 6.3. Based on a Bayesian game, a game theoretic approach is used to develop a strategic platform design game with incomplete information for families of products and services in Sections 6.4 and 6.5.

Chapter 7 presents two examples to demonstrate the usefulness and applicability of the SMPDM for determining a platform design strategy in dynamic and uncertain e-market environment. The SMPDM is employed step-by-step to two case studies
involving a power tool family and a checking account service family to provide proof of concept that it work.

Chapter 8 is the final chapter in the dissertation and contains a summary of the work in Section 8.1. Contributions, limitations, and future work are discussed in Sections 8.2 and 8.3. Finally, closing remarks are given in Section 8.4.
Chapter 2

Literature Review

This chapter reviews literature related to the research areas pertinent to develop the proposed Strategic Module-based Platform Design Method (SMPDM). Approaches for product family and platform design are described via examples in Section 2.1. Applications for family design in products and services are presented in Section 2.2. In Sections 2.3 through 2.5, tools related to the SMPDM are reviewed: (1) design knowledge and data mining, (2) multi-agent system applications, and (3) game theoretic approaches for product and service design, respectively. Current limitations in designing families of products and services are discussed in Section 2.6. Based on this literature review, a discussion of how to extend the concepts from product family design to service family design is given in Section 2.7. Section 2.8 summarizes of this chapter and previews the next.

2.1 Approaches to Product Family and Platform Design

There are two recognized approaches to product family design [80]: (1) a top-down (proactive platform) approach and (2) a bottom-up (reactive redesign) approach. In the top-down approach, a company’s strategy provides guidelines for developing a family of products based on a product platform and its derivatives. For instance, Sony has strategically developed its Walkman® product based on product platforms and derivatives
Meanwhile, the bottom-up approach is focused on redesigning and/or consolidating a group of distinct products to standardize components for sharing and reusing. For example, Black & Decker reduced variety in their products by redesigning their motors.

In platform-based product development, two common types for product families are: module-based product family and scale-based product family. Products in a module-based product family are created by adding, substituting, and/or removing one or more modules from the platform. In a scale-based product family, products are developed by scaling one or more variables related to the platform design to satisfy a variety of market niches. Table 2.1 summarizes examples of both approaches.
Table 2.1: Examples of Module-based and Scale-based Product Families

<table>
<thead>
<tr>
<th>Approach</th>
<th>Company</th>
<th>Product</th>
<th>Features</th>
<th>Reference</th>
</tr>
</thead>
</table>
| Module-based      | Sony                           | Walkman® models              | - Key modules and platforms  
                        |                                 |                              | - Modular design  
                        |                                 |                              | - 250+ model                         | [74]       |
|                   | Nippondenso Co. Ltd.          | Automotive components        | - A combinatoric strategy that involves several different module with standardized interfaced  
                        |                                 |                              | - 288 types of panel motors                         | [93]       |
|                   | Hewlett Packard               | Ink jet and laser jet printers | - Module components to gain benefits of postponing the point differentiation | [20]       |
|                   | Ball Engineering Structure    | Controlled structures        | - One basic modular component  
                        |                                 |                              | - Various shapes and sizes  
                        |                                 |                              | - Customized with options, attachments, and finishes to fit into any size structure | [66]       |
| Scale-based       | Boeing                         | Commercial airplanes         | - Stretching the aircraft  
                        |                                 |                              | - Accommodate more passengers  
                        |                                 |                              | - Carry more cargo  
                        |                                 |                              | - Increase flight range                         | [72]       |
|                   | Honda                          | Automobile platform          | - Stretched in both width and length to realize a “world” car                         | [59]       |
|                   | Rolls Royce                   | RTM322 aircraft engines      | - Scaled by a factor of 1.8 to realize a family of engine  
                        |                                 |                              | - Different Shaft Horse Power and thrust                         | [71]       |
|                   | Black & Decker                | Universal motors             | - Response to new safety regulations  
                        |                                 |                              | - Apply to 122 basic tools with hundreds of variations                         | [43]       |

2.2 Designing Families of Products and Services

A successful product family depends on how well the trade-offs between the economic benefits and performance losses incurred from having a platform are managed.
Simpson, et al. [80] introduced a method to optimize a platform by minimizing performance loss and maximizing commonality. Gonzalez-Zugasti, et al. [24] designed platform modules to minimize design risk and reduce costs relating to developing a product family. Moore, et al. [56] used conjoint analysis to help determine a product platform. Siddique and Rosen [77] described a method to design a platform from an existing group of products by comparing commonalities in assembly processes. Rai and Allada [67] used a two-step approach to determine a modular platform for a product family, which consists of an agent-based optimal technique and post-optimization analysis using the quality loss function.

Meyer and DeTore [52] proposed a platform-based approach to develop new services using methods and processes for applying product family and product platform design and applied the approach to define a new service platform in an international insurance company. Jiao, et al. [32] discussed how design theories and methodologies for products and manufacturing systems can be applied to the design of service delivery systems for mass customization. They considered a service delivery system as a product system instead of an operational system. Perters and Saidin [65] investigated key factors for the implementation of service customization and used service modules to represent the levels of modularization of the scope of work and design process. Li [45] introduced concepts and presumptions for service package and service product module level in service innovation and service product development.
2.3 Design Knowledge and Data Mining

Design knowledge is considered as the collection of knowledge that can support the design activities and decision-making in product development [100]. As the amount of information related to design and the complexity of products increases, knowledge management systems face the challenge of supporting designers to find appropriate information and are difficult to control with human resources [49, 87]. Artificial Intelligence (AI) techniques and information technology (IT) provide a natural means to facilitate knowledge management for performing knowledge acquisition, knowledge repositories, knowledge discovery, and knowledge distribution [48]. Zha and Sriram [100] proposed a knowledge-intensive support system for capturing, representing, and managing design knowledge based on platform-based product family design, and proposed a system implementation architecture and functionality. Mulet and Vidal [57] introduced functional requirements for developing knowledge-based design support systems through an experimental study using a FBS (function, behavior, structure) model and Linkography to represent and analysis design processes. Chau [14] presents an ontology-based knowledge system to develop a mathematical model for flow and water quality by sharing, reasoning, and managing domain knowledge.

Various data mining approaches have been developed to support product and product family design. Agard and Kusiak [1] proposed a three-step methodology for the design of product families based on the analysis of customers’ requirements using a data mining approach. In the first step, data mining algorithms are used for customer segmentation. The second step provides a function structure to satisfy the diversified
requirements. A product structure and distinguished modules for the product variability are designed in the final step. Jiao and Zhang [33] presented a domain-independent inference system for analyzing and organizing requirement information to support product portfolio identification. They identified association rules to provide an integration of requirement information from both customer and design viewpoints within a coherent framework. Xue and Dong [97] introduced the coding and clustering of design and manufacturing features for concurrent design. They employed a fuzzy pattern clustering algorithm to organize a large design feature library into hierarchical feature groups. Liao [47] used a classification and coding approach to solve the part family formation problem in group technology applications. He defined fuzzy design features, mapping them to the code structure, and applied similarity-based clustering methods to form part families. Romanowski and Nagi [70] presented a data mining-based methodology to form a generic bill of materials for variant design using text mining and association rule mining.

An association rule mining technique can help find interesting associations or correlation relationships among a large set of data items [33]. The results of association rule mining can be design knowledge that is used to define a platform and common modules. Clustering can be used to group customers or functions of similar behavior [1, 33]. Also, functional requirements in existing products can be clustered based on the similarity between them. This process can be achieved by clustering algorithms such as the k-means algorithm, hierarchical algorithms, pattern recognition, Bayesian statistics, neural networks, and support vector machines [1]. Agard and Kusiak [2] used a clustering method to identifying similar customers in the design of standardized product. They
applied a neural network to learn customers’ preferences to build a target customer and cluster customers’ behaviors. Lozano, et al. [50] proposed a modified fuzzy c-mean clustering algorithm for part-machine grouping in cellular manufacturing. They considered the link between the degree of membership of machines and part families, and the effect of the weighting exponent on the fuzziness of the solution.

2.4 Multi-Agent Systems and Their Applications

Software agents provide an ideal mechanism to integrate information during design and manufacturing. Madhusudan [51] developed a flexible agent-based coordination framework for new product development in a distributed design process system. Jia, et al. [31] presented an agent-based system for coordinated product development and manufacturing that is able to execute tasks in a coordinated and flexible way. Anumba, et al. [5] introduced a multi-agent system framework to facilitate collaborative design and interaction protocols for agent negotiation and applied it to design industrial buildings. Tan, et al. [88] developed a multi-agent framework to provide information that helps designers, engineers, and managers work together to improve initial designs by satisfying a wider variety of concerns.

In a distributed environment, agent-based market mechanisms can provide a solution for various applications with increasing information and complexity related to decision-making. Reeves, et al. [68] applied a market-based scheduling mechanism to derive strategies for bidding agents in an allocation problem. Lee, et al. [41] introduced a multi-agent-based information infrastructure model that facilitates a simulator based on a
market mechanism, P-TÂTO, and applied the model to solve a resource scheduling problem. Blecker, et al. [11] proposed a multi-agent-based configuration process to find the optimal product variety for module-based products using a market mechanism that consists of the target costing concept and a Dutch auction.

In agent-based electronic markets, reputation is often used to detect and dismiss fraudulent agents [63, 91]. Zacharia, et al. [98] presented a framework for agent-mediated knowledge marketplaces in which agents’ reputations are established by dynamic pricing algorithms. Padovan, et al. [63] described a prototypical implementation of an automated subsequent treatment of reputation information in a multi-agent system. Tran and Cohen [91] proposed a reinforcement learning and reputation-based algorithm for buyers and sellers in agent-based electronic marketplaces. The algorithm was focused on maximizing the expected value of goods for buyers, and maximizing the expected profit for sellers.

2.5 Game Theoretic Approaches

Game theoretic approaches provide a rigorous framework for managing and evaluating strategies to achieve players’ goals using their complete or incomplete information and knowledge [23]. A game is a description of strategic interaction that includes constraints based on players’ actions. Game theory provides reasonable solutions for various games and evaluates their properties [23, 62]. According to constraints and the situations of games, game theoretic models can be partitioned into three categories: (1) cooperative and non-cooperative games, (2) strategic and extensive games, and (3) games with complete and incomplete information [62]. In engineering design, game
theoretic approaches have been applied to model strategic relationships between
designers for sharing design knowledge and solving design problems. Xiao, et al. [96] applied game theoretic approaches and design capability indices to model the relationships between engineering teams that were described as cooperative, non-cooperative, and leader/follower protocols, and facilitate collaborative decision making during a product realization process. Fernandez, et al. [21] proposed a framework for establishing and managing collaborative design spaces by combining elements of cooperative and non-cooperative behavior, and formulating strategic and extensive games with utility theory. Kopin and Wilbur [38] introduced a Bayesian game to model cost sharing in uncertain and incomplete information that were related to producer and consumer attributes such as nature, production costs, players and information, and preferences. Correia [16] investigated the representation of incomplete and asymmetric information to model a strategic Bayesian game that was represented by the constraints of a transmission system and player’s strategic reactions to estimate uncertainties. Lewis and Mistree [44] presented mathematical constructs for modeling a multidisciplinary design optimization problem using game theoretic principles and the compromised Decision Support Problem (DSP) in a collaborative, sequential, and isolated design environment. Lariviere and Van Mieghem [40] used a timing game to determine customers’ arriving behaviors based on a delay cost function in a service facility and proposed appropriate strategies to select arrival times for the various volumes of service capacities. Finally, Huang, et al. [29] described a multi-stage non-cooperative configuration game between platform products and supply chains to determine the optimal configuration decisions of a manufacturer and suppliers for mass customization,
and demonstrated the applications of the proposed game and solution procedure using a series of simulation experiments and a numerical example.

2.6 Limitations of Current Methods for Designing Families of Products and Services

When developing a product family, effective design strategies should be built upon a company’s experience and/or their product information to minimize product development risk and reduce time to market. The bottom-up approach can provide design knowledge for this process by extracting information from existing products. Therefore, as shown in Figure 2.1, we need a design method to combine and analyze the design knowledge and strategies simultaneously. Multi-agent systems (MASs) can provide a suitable method to integrate distributed design knowledge and information. Data mining techniques can be used to manage existing product information and discover knowledge related to design and supported by MASs. Agent-based decision-making can also provide more consistent methods to determine design factors in distributed and dynamic design environments.
platform design is essential for successful family design as demonstrated in numerous products and services; however, the majority of previous research in family design has been focused on developing products and services in a deterministic manufacturing environment. In product family design for mass customization, a method to produce a variety of products should consider dynamic and various market segments to incorporate a variety of customer needs and trends. In addition, dynamic factors, like designing customer needs and trends, companies’ strategies, and technologies, should be considered to increase customer’s satisfaction when developing a family of products and services. Therefore, we need to address how to capture such dynamic factors when designing a family of services as well as products. Market-based product design is one way to reflect various and dynamic market environments in product design. MASs can
help achieve higher levels of flexibility, scalability, and adaptability in dynamic and distributed environment [41]. A multi-agent approach can be applied to develop an appropriate method to model a market-based product design system for the following reasons: (1) multi-agent systems and market-based mechanisms are inherently distributed, (2) products can be designed with modules whose information is distributed across a market, (3) each module can be modeled as a self-interested agent – an autonomous decision-maker and a specific information holder at the same time, and (4) a simple market mechanism can reduce the amount of communication in the distributed negotiation environment so that it increases the overall effectiveness in a product design system. The multi-agent approach can be extended to develop market-based service design systems as well as demonstrated in this work in Chapter 6.

In the literature review, the two platform design approaches have proven to be successful in product family design; therefore, the two approaches can be extended to develop families of services. For example, module-based approaches can help design a service family consisting of processes, functions, or optional modules. Meanwhile, scalability is an important factor to create a variety of services in service design. Scale-based approaches can provide appropriate methods to develop a service family applied to scaleable services within a family.

2.7 Extensions to Service Design

In the literature review, most research is focused on increasing a variety of services by developing families of services; however, there is minimal research for designing a
platform for these families. Therefore, this research focuses on how to design a service platform for the effective design of families of services by extending concepts from product family design and using agent-based decision-making.

Based on the literature review, the following definitions for a service family and a service platform are proposed:

- **A service family** is a set of services based on a service platform, facilitating mass customization by promoting customer value and providing a variety of services for different market segments cost-effectively.

- **A service platform** is a common basis that consists of processes, activities, objects, and/or features that are shared and remain constant from service to service, within a given service family.

- **A service module** is a set of service components for performing a service.

- **A service component** is regarded as an activity to satisfy certain services, which are defined by a set of processes, operations, people, objects, and/or features.

These definitions are novel to this work and provide a foundation for extending product family design to service family design. Table 2.1 shows possible methods for designing a family of services and limitations based on the literature review. Two recognized approaches, *platform-based* and *module-based*, are used to provide a variety of services in a family of services. A platform-based approach can be realized by a top-down approach for product family design. A module-based approach is similar to a bottom-up approach for the product family design when developing service modules.
Therefore, it is possible to extend concepts from the product family design to service platform and family design.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Method</th>
<th>Limitation</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Platform-based</td>
<td>Define platform groups as several teams that perform core service activities</td>
<td>Consider only service activities</td>
<td>[52]</td>
</tr>
<tr>
<td></td>
<td>Consider services as operational products</td>
<td>Discuss the possibility for service family design</td>
<td>[32]</td>
</tr>
<tr>
<td>Module-based</td>
<td>Define modularization level based on service process hierarchy</td>
<td>Develop a variety of services by the combination of modulus</td>
<td>[65]</td>
</tr>
<tr>
<td></td>
<td>Configure service package that consists of service product content and process</td>
<td>Propose some concepts and presumptions</td>
<td>[45]</td>
</tr>
</tbody>
</table>

In this research, we extend concepts from platform-based product design to develop a module-based service family as shown in Figure 2.2. A service platform consists of common service modules that are defined as service components representing processes and capabilities. Based on the service platform, we can create a variety of services - and ultimately families of services - for satisfying various market segments depending on factors such as location, facility design and layout for effective customer and work flow, procedures and job definitions for service providers, measures to ensure quality, extent of customer involvement, equipment selection, and adequate service capacity [22].
2.8 Summary and Preview

In this chapter, a literature review and current limitations related to developing the SMPDM were presented. A discussion of how to extend concepts from product family design methods to service family design was described by providing definitions and design strategies with respect to a service family. In the next chapter, the tools reviewed in this chapter are integrated to create the SMPDM for determining an optimal platform design strategy. Steps in the SMPDM are presented with examples. Research objectives and the validation strategy for testing the SMPDM are discussed, including case studies and scenario-based experiments to verify the objectives in the remainder of this dissertation.

Figure 2.2: Process of Service Family Design and Service Families
Chapter 3

The Strategic Module-based Platform Design Method (SMPDM)

This chapter introduces the Strategic Module-based Platform Design Method (SMPDM) for determining a platform strategy to support product and service family design in dynamic and uncertain market environments. Section 3.1 introduces the SMPDM, and details of the constituent tools are presented in Sections 3.1.1 through 3.1.4 based on the foundations reviewed in Chapters 1 and 2. An outline of the strategy for verification and testing of the SMPDM is discussed in Section 3.2. Section 3.3 concludes the chapter with a recap of what has been presented and previews the next chapter.

3.1 The Strategic Module-based Platform Design Method

As mentioned in Section 1.3.3, the principal contribution in this dissertation is the new Strategic Module-based Platform Design Method (SMPDM) for determining a platform strategy to support product and service family design in dynamic and uncertain market environments. The proposed method extends concepts from product family and platform design into service design and is focused on a bottom-up approach and module-based design. Data mining techniques are used to identify a platform and corresponding modules by investigating design information and knowledge in a large database or repository. Agent-based decision-making is applied to determine module-based platform
design in dynamic and uncertain market environments. The steps and associated tools in the SMPDM are illustrated in Figure 3.1.

![SMPDM Steps Diagram](image)

**Figure 3.1: Steps and Tools of the SMPDM**

The proposed method consists of four steps: 1) analyze design data, 2) represent design knowledge 3) identify modules and strategies, and 4) specify platform strategy. The first step is to analyze overall design information and constraints related to customer needs and functional requirements for the products and/or services. In the second step, design knowledge is represented using an ontology. Modules and platform strategies are identified by data mining techniques and the fuzzy set theory at the third step. The fourth step is to determine an optimal platform strategy using agent-based decision-making.
algorithms. Figure 3.2 shows relationships between steps in the SMPDM and the factors for developing a multi-agent decision making framework. Details on each step follow.

3.1.1 Step 1 – Analyze Family Design Data

Given the overall design information and constraints, Step 1 in the SMPDM is to obtain and understand data needed to develop platform strategies during the development of families of products and services. Figure 3.2 shows the proposed process of developing a family of products and services based on customer needs (CNs). Information required to identify CNs can be collected by surveying prospective customers and by conducting a marketing study that begins by establishing target market
segments and customers. In the initial phase, CNs are analyzed to understand customer intention and determine a strategy for developing a product and service family. For example, the number of products can be decided by customer groups classified according to CNs. CNs are also used to identify appropriate functional requirements (FRs), which are then mapped to FRs [85]. For a product, FRs describe a product’s behavior and features that are defined by technical information and data for its design. In service design, FRs represent processes and capabilities that are determined by work flow, procedures and job definitions for service providers, and service quality. In the conceptual design phase, products and services can be designed based on FRs, and their functional modules can also be determined. In particular, a family of products and services are first configured by defining a platform. A platform consists of the common modules that can be shared across the family of products and services. After conceptual design through prototype and pre-production, final products and services are manufactured and delivered, respectively.

Figure 3.3: The Process of Developing a Family of Products and Services
As discussed in Section 2.1.1, several product family design methods provide solutions to analyze information in databases or design repositories for supporting design knowledge representation. Regarding product data analysis, a functional model can help decompose a product into functional modules and components based on predefined terms in product design. For example, using the information in a design repository that consists of the bill of materials, assembly relationship, functional flows, and energy flows for products, function structure models can be developed for each product. Then, based on the functional structure models, modules for the products are defined by the heuristic method of Stone and Wood [25]. Figure 3.3 shows the functional model for a circular saw and its five modules (refer to Appendix A).

![Functional Model of a Circular Saw and Five Modules](image-url)
3.1.2 Step 2 – Represent Design Knowledge

Based on information derived from Step 1, Step 2 of the SMPDM develops representations for products and services to support sharing and reusing design information using ontologies and object-oriented concepts. Representing products and services by ontologies facilitates the application of data mining techniques by capturing, configuring, and reasoning both linguistic and parametric design information in a knowledge management system effectively [58]. For example, Figure 3.4 shows the functional hierarchy for a drill as an example along with the hierarchy for representing a product using an ontology. In Chapter 4, the process of representing design knowledge for a product and service family is described in detail.

![Figure 3.5: Functional Hierarchy and Product Representation using an Ontology](image-url)
3.1.3 Step 3 – Identify Modules and Platform Strategies

Once the representation of design knowledge has been constructed, Step 3 in the SMPDM identifies modules and platform strategies. Clustering methods are used to determine modules by measuring the similarity of functional features for products and services. Classification and association rule mining are also employed to support the identification of modules that result from clustering.

Modules resulting from clustering can be categorized based on function into: 1) unique, 2) common, and 3) variant modules. Unique modules are based on distinct functions within a family of products and services so that components in the modules cannot be replaced by those in the different modules to fulfill their task. Common modules are based on common functions within the family so that components in the modules can be shared. Variant modules are based on common functions but differ in one or more of their attributes. Variant modules can become unique modules or common modules according to the results of trade-off analysis between modules’ performance and commonality. The platform consists of the common modules and the variable modules that can be common modules. More detail for these trade-off relationships are discussed in Section 6.1.1.

Based on these trade-off relationships, designers can identify a feasible set of strategies for the platform of products and services. The strategies are represented as alternative functional combinations for a platform and can be constructed by combining variant modules along with common modules based on functional features in the family. Fuzzy set theory is used to create various platform strategies based on designers’
knowledge that are represented by linguistic design information. In Chapter 5, the process of identifying modules and a platform for a product and service family is described in detail. Alternative platform design strategies are evaluated through agent-based decision-making, and an optimal platform strategy is determined as discussed in Chapter 6.

3.1.4 Step 4 – Specify Platform Design Strategy

Step 4 in the SMPDM uses agent-based decision-making to determine an optimal platform strategy for product and service family design in dynamic and uncertain environments. Based on Figure 3.2, a multi-agent system (MAS) is proposed for applying the agent-based decision-making to specify platform strategies, as shown in Figure 3.5. The roles and knowledge of each agent are described in Table 3.1. Decision-making methods in the MAS are determined by agents’ roles and module’s characteristics. Based on the literature review, Table 3.2 shows suitable decision-making methods to determine a platform and modules in a dynamic market environment. In Chapter 6, we address how a MAS can be used to determine an optimal platform strategy for designing a family of products and services in dynamic and uncertain market environments, and describe the methods related to agent-based decision-making in detail.
Figure 3.6: Multi-Agent System Architecture based on a Dynamic Market Environment
<table>
<thead>
<tr>
<th>Agent</th>
<th>Roles</th>
<th>Knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer Agent (CTA)</td>
<td>• Interface with customers</td>
<td>• Customer needs</td>
</tr>
<tr>
<td></td>
<td>• Analysis of customer needs (CNs)</td>
<td>• Inference algorithm for CNs</td>
</tr>
<tr>
<td></td>
<td>• Request to a coordinator agent</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Map CNs to FRs</td>
<td></td>
</tr>
<tr>
<td>Product, Service Agents (PDA)</td>
<td>• Analysis Functional requirements</td>
<td>• Functional requirements</td>
</tr>
<tr>
<td></td>
<td>• Evaluation for existing products</td>
<td>• Inference algorithm for FRs</td>
</tr>
<tr>
<td></td>
<td>• Evaluation metrics (performance and commonality)</td>
<td>• Evaluation metrics (performance and commonality)</td>
</tr>
<tr>
<td>Coordinator Agent (CDA)</td>
<td>• Task decomposition (a product or service into modules)</td>
<td>• Design methods</td>
</tr>
<tr>
<td></td>
<td>• Definition of Module Agent Techspecs</td>
<td>• Decomposition algorithm</td>
</tr>
<tr>
<td></td>
<td>• Task (resource) allocation</td>
<td>• Component information related with FRs</td>
</tr>
<tr>
<td></td>
<td>• Management for PFA, VDA, PPA, MDAs</td>
<td>• PFA, VDA, PPA, MDAs information</td>
</tr>
<tr>
<td></td>
<td>• Decomposition algorithm</td>
<td>• Techspecs information</td>
</tr>
<tr>
<td>Platform Agent (PFA)</td>
<td>• Platform design</td>
<td>• Platform design methods</td>
</tr>
<tr>
<td></td>
<td>• Decision making – module selection for platform</td>
<td>• Strategy for negotiation</td>
</tr>
<tr>
<td></td>
<td>• MDAs information</td>
<td>• MDAs information</td>
</tr>
<tr>
<td>Variant Design Agent (VDA)</td>
<td>• Product and service family design</td>
<td>• Design information</td>
</tr>
<tr>
<td></td>
<td>• Evaluation for new or redesign products</td>
<td>• Family design methods</td>
</tr>
<tr>
<td>Production Planning Agent (PPA)</td>
<td>• Production and process planning for manufacturing system, service</td>
<td>• Product and service design information</td>
</tr>
<tr>
<td></td>
<td>• MDAs information</td>
<td>• Production planning algorithms</td>
</tr>
<tr>
<td>Module Agents (MDA)</td>
<td>• Decision-making - module design</td>
<td>• Module design methods</td>
</tr>
<tr>
<td></td>
<td>• Performance subtasks</td>
<td>• Module information (Techspecs)</td>
</tr>
<tr>
<td></td>
<td>• Request to CPAs and PMAs</td>
<td>• Strategy for negotiation</td>
</tr>
<tr>
<td></td>
<td>• Module reputation update</td>
<td>• Learning algorithm</td>
</tr>
<tr>
<td></td>
<td>• Evaluation for the quality of modules and components</td>
<td>• PMAs reputation</td>
</tr>
<tr>
<td></td>
<td>• MDAs, PMA, CPAs information</td>
<td>• CPAs information</td>
</tr>
<tr>
<td>Market Manager Agent (MMA)</td>
<td>• Market management</td>
<td>• MDAs, PMAs, and CPAs information</td>
</tr>
<tr>
<td></td>
<td>• Tracking the coming and leaving agents</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• MDAs, PMAs, and CPAs information records</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Connection MDAs, PMAs, and CPAs</td>
<td></td>
</tr>
<tr>
<td>Product Market Agents (PMA)</td>
<td>• Evaluation of products in market</td>
<td>• Product and service information</td>
</tr>
<tr>
<td></td>
<td>• Customer satisfaction evaluation for services</td>
<td>• Customer needs and satisfaction</td>
</tr>
<tr>
<td>Component Agents (CPA)</td>
<td>• Management repository</td>
<td>• Component information</td>
</tr>
<tr>
<td></td>
<td>• Component search</td>
<td>• Searching algorithm</td>
</tr>
<tr>
<td></td>
<td>• Repository updating</td>
<td>• Strategy for negotiation</td>
</tr>
</tbody>
</table>
3.2 Strategy for Verification and Testing of the SMPDM

When proposing a new design method, validation is necessary to guide the development and evaluation of the method. To perform an evaluation of decision support methods, Olewnik and Lewis [61] propose three elements for validation based upon reason and concepts in the existing literature. For a decision support method to be valid, it must:

1. *Be logical*

2. *Use meaningful, reliable information*

3. *Not bias the designer*

For design methods, Antonsson [4] introduces a definition as follows:

<table>
<thead>
<tr>
<th>Agents</th>
<th>Method</th>
<th>Decision</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDA</td>
<td>Data mining (association rule, clustering, classification)</td>
<td>Task decomposition (modules)</td>
<td>[1, 12, 33, 41]</td>
</tr>
<tr>
<td>PFA – MDAs</td>
<td>Negotiation, Game theory</td>
<td>Platform for product or service</td>
<td>[5, 75, 94]</td>
</tr>
<tr>
<td>VDA – MDAs</td>
<td>Collaborative negotiation</td>
<td>Module for product or service</td>
<td>[46, 51, 75, 94]</td>
</tr>
<tr>
<td>MDAs – PMAs</td>
<td>Learning mechanism</td>
<td>Module for product or service</td>
<td>[63, 91, 98]</td>
</tr>
<tr>
<td>MDAs – CPAs</td>
<td>Combinatorial auction</td>
<td>Module for product or service</td>
<td>[95]</td>
</tr>
<tr>
<td>VDA</td>
<td>Data mining, family design methods</td>
<td>Configuration modules</td>
<td>[12, 33, 35, 77, 81]</td>
</tr>
<tr>
<td>PPA</td>
<td>Production planning, design and manufacturing methods</td>
<td>Production and process schedule</td>
<td>[81, 88]</td>
</tr>
</tbody>
</table>
“The validation of design methods is achieved ultimately by results and usefulness, and through a convincing demonstration to one’s peers in the field.”

Based on these definitions, a validation strategy for this research is established as follows: 1) comparison of logical steps, 2) utilization of current design information, 3) assessment of applicability, and 4) evaluation of usefulness.

According to the strategy for validating the SMPDM, testing each step in the SMPDM is described as follows. The usefulness and applicability of the tools in the SMPDM were introduced through literature review in Chapter 2. Further verification of the SMPDM requires demonstrating that these tools can be incorporated into the SMPDM. Table 3.3 shows the relationships between the next four chapters and the steps in the SMPDM.

<table>
<thead>
<tr>
<th>Step</th>
<th>Tool</th>
<th>Chp. 4</th>
<th>Chp. 5</th>
<th>Chp. 6</th>
<th>Chp. 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>Product Family Design Methods</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Step 2</td>
<td>Ontology</td>
<td>×</td>
<td></td>
<td></td>
<td>×</td>
</tr>
<tr>
<td>Step 3</td>
<td>Data Mining</td>
<td></td>
<td>×</td>
<td></td>
<td>×</td>
</tr>
<tr>
<td>Step 4</td>
<td>Multi-agent System</td>
<td></td>
<td></td>
<td>×</td>
<td>×</td>
</tr>
</tbody>
</table>

To validate the SMPDM, case studies and multiple scenario-based experiments are presented to demonstrate its applicability and usefulness based on existing products and services: (1) platform design of a family of power tools, (2) platform design of a family of checking account services, and (3) scenarios reflecting a dynamic and uncertain market environment. The case studies and the scenario-based experiments are used to demonstrate the following (with relevant SMPDM’s steps and chapters noted in
• Platform design of a family of power tools
  o Representing design knowledge for products (Steps 1&2, Chapter 4),
  o Identifying modules and platform strategies (Step 3, Chapter 5),
  o Discovering design knowledge (Step 3, Chapter 5),
  o Implementing a market-based negotiation mechanism (Step 4, Chapter 6), and
  o Determining an optimal platform strategy in a dynamic and uncertain market environment (Step 4, Chapters 6 and 7).

• Platform design a family of checking account services:
  o Presenting design knowledge for services (Steps 1&2, Chapter 4),
  o Identifying modules and platform strategies (Step 3, Chapter 5), and
  o Determining an optimal platform strategy in a dynamic and uncertain market environment (Step 4, Chapters 6 and 7).

• Scenario-based implementations related to an electronic market:
  o Implementing a negotiation mechanism and a Bayesian game (Step 4, Chapters 6 and 7).
  o Determining a platform design strategy in an electronic market environment (Step 4, Chapters 6 and 7).
3.3 Summary and Preview

The SMPDM for determining platform strategies to support product and service family design in dynamic and uncertain market environments has been introduced. Detailed constituent elements of the SMPDM are elaborated. In summary, the following steps constitute the SMPDM:

- Step 1 – Analyze market and design data
- Step 2 – Represent design knowledge for products and services
- Step 3 – Identify modules and platform design strategies
- Step 4 – Specify platform design strategy

The SMPDM can help yield an effective platform design strategy in dynamic and uncertain market environments. The next chapter provides design knowledge representation for families of products and services using ontologies.
Chapter 4

Design Knowledge Representation using an Ontology

This chapter provides the foundation for representing products and services using an ontology. Ontological representations can help represent design knowledge for products and services as well as support data structures for applying decision-making methods in the SMPDM to improve the process of searching for components satisfying particular customer needs and functional requirements. This chapter is organized as follows. An overview of ontologies is presented in Section 4.1, and the Techspecs Concept Ontology (TCO) is investigated in Section 4.2 to represent product and component knowledge. A case study for a family of power tools is performed using the software Protégé\(^4\) and JARE\(^5\) in Section 4.2.6. A service ontology is then developed by introducing a function-process matrix, object-oriented concepts, and a service process model in Section 4.3. In Section 4.3.4, a family of checking account services is as a case study for the service ontology. A summary of this chapter and a preview of the next are discussed in Section 4.4. Figure 4.1 shows the roadmap for this chapter.

\(^4\) http://protege.stanford.edu
\(^5\) http://jare.sourceforge.net
4.1 Overview of Ontologies in Product Knowledge Representation

Companies are making increased efforts to reduce cost and time for developing new products to survive in today’s competitive market. To achieve such goals, it is essential to provide the flexibility and responsiveness needed for product development process; however, many resources are being spent examining large collections of corporate legacy data for designs which may be modified to solve new problems, or searching through catalogs for components to be incorporated into new designs [92]. Sharing and reusing product design information can help eliminate such wastes and facilitate product family design. To share and reuse design information, it is important to adopt an appropriate representation scheme for components and products. The objective
in this chapter is to develop a method for representing products and services using an ontology.

An ontology consists of a set of concepts or terms and their relationships that describe some area of knowledge or build a representation of it [86]. Ontologies can be defined by identifying these concepts and the relationships between them and have simple rules to combine concepts for a particular domain. Using an ontology, we can capture domain knowledge in a generic way and provide a common vocabulary for a domain that may be shared and reused among teams and even software agents. Generally, we can consider developing a descriptive ontology as defining a set of data and its structure which is analogous to formulating a generalized information flow model for agents to use.

Ontologies have proven to be useful in application areas such as intelligent information integration, information brokering, and knowledge-based systems [82]. Mohan and Ramesh [55] developed an ontology that catalogues different concepts associated with variability. This ontology was used to define the elements characterizing the knowledge elements necessary for managing variability in product/service families. They have also developed a knowledge management system integrated with an ontology development tool to facilitate knowledge capture and retrieval for variability management. Kitamura, et al. [37] proposed a functional concept ontology that provides a rich vocabulary representing functions together with clear definitions grounded on behavior. Nanda, et al. [58] applied ontologies to represent products and product families for promoting component sharing and reuse and assisting designers search, explore, and
analyze linguistic and parametric product family design information in a knowledge management system effectively.

4.1.1 Device ontology

A device ontology is developed to provide a proper form that can successfully model a product with consistency in various domains [54]. A device-centered view of artifacts provides the basic concept of a device ontology. Any artifact or component is considered as a composition of devices and a main actor (agent) in an environment in which input and output change. Operands are defined as things that change the input and the output. In other words, a device changes states of things inputted, which are called operands such as substances like fluid, energy like heat, motion, force, and information. Pahl and Beitz [64] suggested the basic idea of a device ontology composed of components and relationships between them in many system engineering areas. De Kleer and Brown [18] proposed a concept of a conduit in the behavior of a physical system. Conduit is defined as a device that transports materials from one component to another without changing any feature of the material within them. Mizoguchi and Kitamura [54] extended the device ontology to include various behaviors and the concept of a medium. A medium is something that holds an operand and enables it to flow among devices. In this work, we employ the device ontology to capture functions and relationships in products consistently. Figure 4.3 shows the overall function of a circular saw’s main components, and the table in Figure 4.3 is its corresponding ontology represented as a
device ontology. The circular saw has a disc-like circular blade attachment that is used to saw through blocks and sheets.

4.2 Techspecs Concept Ontology (TCO)

In this section, we describe the Techspecs Concept Ontology (TCO) which consists of functionality, assembly, and technical specification of products. We use a device ontology for representing the behavior, functionality, and relationships of products and their components. In particular, relationships between components are defined by an assembly relationship. All components with a function relation and an assembly relation have technical specifications with specific formats derived from its functional requirements. Based on this information, we develop knowledge for product design.

As shown in Figure 4.2, TCO for representing components consists of a functional level, an assembly level, and a product technical specification. The lower layer defines and provides very basic concepts such as products, platforms, and components. A device
ontology is employed to provide a common viewpoint that represents the consistent interpretation of artifacts. The functional level describes a functional concept as an instance of the concept of ‘function’ defined in the device ontology [36]. The assembly level specifies assembly relationships between components or modules as an instance of the concept of ‘assembly’ defined in the physical process and device ontology. The relation between function, assembly, and connection can be defined based on the modules in a product. Product technical specifications represent the specific information or knowledge based on the lower layer, which is derived from functional requirements that consist of component and assembly specifications. TCO can help provide a designer (or agent) with a feasible solution to make an initial product from the selection of components for the assembly relation. We elaborate on the details in the following sections using a circular saw as an example.

![Techspecs Concept Ontology Diagram](image)

**Figure 4.3: Hierarchy of Techspecs Concept Ontology**
4.2.1 Functional Level

Fundamental and generic concepts for capturing and describing the functional knowledge should be defined as consistent and sharable descriptions. Generally such specification of a conceptualization is called an “ontology” [54]. Using both the behavior relations and the functional relations of a product or components, we can develop its functional level based on the device ontology. In the conceptual design phase, a designer can use the functional level to propose an initial product satisfying functions based on customer needs.

A behavioral relation of a system is defined as a set of model of components (devices) that represent the behavior of each component, relation information among behaviors of the components, and the structural hierarchy of the related components. A component can have several behavioral forms. Mizoguchi and Kitamura [54] define “behavior” of the component as the difference between the states of the operand at the input and the output in a system.

We can use a functional decomposition to represent the intended behavior (i.e., the functions) of a product and its components. A function can be defined as a single physical component or a combination of components described in specific technical specifications or processes [35]. Functional components are arranged based on a number of logical intentions that ensures the fulfillment of the functional objective. The logical arrangement is called a working principle, which specifies the mode of action that the product or system carries out on the inputs to achieve the output state [35]. In a functional model, the function flow diagram provides the information of how the function is being
implemented in the system and how each component supports the product’s overall function [35].

The functional modeling procedure presented by Stone, et al. [83] is applied to generate a graphical representation of decomposed sub-functions. All the sub-functions are connected through the flow of energy, material, and/or signals to form an organized structure. A formal and systematic methodology of describing the component function is fundamental when creating a function model structure. Figure 4.4 shows the functional model for the circular saw as an example.

As shown in Figure 4.4, a solid box defines each sub-function of the product and identifies all flows in and out of the product. A functional model is then created using the functional basis [83]. Each flow is traced through the product, and each function that the product performs on each flow is listed in verb-noun format. Some component functions in a functional model are combined into a module representing a modular function. This
is indicated by the dashed boxes in Figure 4.4. Based on a module, we can develop the functional hierarchy of a product, like a functional decomposition tree. Figure 4.5 shows the functional hierarchy for the circular saw.

To represent a function level in TCO, we utilize properties called slots in the ontology. The table in Figure 4.5 shows several properties for representing the function level. The function of a product represented by TCO provides semantics for a designer to understand the meaning of a function. For example, ‘Operand (force) Transfer’ means that the ‘Transfer’ function has force as input and output.

Figure 4.5: Functional Hierarchy of a Circular Saw and Properties

4.2.2 Assembly Level

A product can be decomposed into sub-systems and/or sub-assemblies capable of achieving the overall product function. The decomposition process should continue until basic physical components are reached [35]. Figure 4.6 shows the physical decomposition for the circular saw and properties for an assembly relation based on an
assembly module. The overall system is decomposed into several physical sub-assemblies that include the motor sub-assembly. Then the motor sub-assembly is decomposed into its basic physical components as shown in Figure 4.6.

In Figure 4.7, the connecting information between components is defined as a conduit. For example, Gear1 and Gear2 have a contact as the conduit. In particular, the connection between a motor and gear train housing is mounted by two bolts. This information is part of a technical specification for a circular saw. Using the physical decomposition hierarchy and connection information, we can develop an ontology related to the product assembly.
Figure 4.8 shows the overall relation of elements for TCO. As shown in Figure 4.8, the line represents relationships between behavior, function, assembly, and connection for specific components. A designer can use this matching information to select appropriate components for satisfying customer needs during conceptual design. Based on this information, we can also develop knowledge that consists of rules and facts for product design. The following are examples of rules and facts for a circular saw.

Rules:
- If connection is between motor and gear1
  Then conduit has shaft
- If connection is between motor and switch
  Then conduit has wire

Facts:
- Function Convert
- Function Transfer
- Convert electronic energy to mechanical energy
- Transfer electronic energy
- Transfer force

![Behavior hierarchy]

Figure 4.8: Overall Relation between Hierarchies
4.2.3 Product Technical Specifications

In general, product technical specifications describe the characteristics of a product or component. In large assemblies such as aircraft, automobiles, and ships, reconciliation of technical specifications with the design specifications is often cumbersome. If these specifications are not correctly reflected in an initial design phase, then this could lead to scrapping the manufactured part and remanufacturing it, increasing cost. Therefore, we need to link computer representations of assemblies and the Technical Requirements Document or Specifications that should lead to their design [15].

In TCO, all components related with the functional level and the assembly level have technical specifications with specific formats for related components. The specific format of a component is derived from its functional requirements. Table 4.1 shows the properties of component specifications based on functional requirements. Products may have two types of specifications in TCO: (1) component specifications and (2) assembly specifications. The assembly specification has an assembly relation and a connection relation. Therefore, TCO provides designers with a product technical specification for use during the early phases of design.

Table 4.1: Properties of Technical Specifications

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Cardinality</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>component name</td>
<td>string</td>
<td>single</td>
<td>name</td>
</tr>
<tr>
<td>component function</td>
<td>instance</td>
<td>multiple</td>
<td>classes = (function)</td>
</tr>
<tr>
<td>weight</td>
<td>string</td>
<td>single</td>
<td>component’s weight</td>
</tr>
<tr>
<td>size</td>
<td>string</td>
<td>single</td>
<td>component’s size</td>
</tr>
<tr>
<td>specification</td>
<td>string</td>
<td>multiple</td>
<td>component or product specification</td>
</tr>
<tr>
<td>material</td>
<td>string</td>
<td>multiple</td>
<td>component’s material</td>
</tr>
<tr>
<td>commonality</td>
<td>symbol</td>
<td>single</td>
<td>allowed-values={Unique, Variant, Common}</td>
</tr>
<tr>
<td>product name</td>
<td>string</td>
<td>single</td>
<td>product name</td>
</tr>
</tbody>
</table>
We can use TCO to represent semantic information for products or components to better reflect customers’ preferences as well as market needs. In addition, assembly relationship information in TCO gives the specific combination of related components. It is possible that a designer can search all related components using some basic functions or behaviors based on customer needs. Technical specifications related components also provide the feasibility and the limitation of the assembly of new product platform. This information plays an important role in conceptual design; therefore, TCO can help develop an effective product platform and product family.

4.2.4 Product Representation

Using TCO, a module-based functional hierarchy for a product can be developed as shown in Figure 4.9. Suppose that a product family consists of \( l \) products, \( PF = (P_1, P_2, \ldots, P_l) \) and a product consists of \( m_i \) modules, \( P_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,m_i}) \), where \( x_{i,j} \) denotes a module \( j \) in product \( i \) and consists of a vector of length \( n_{m} \), \( x_{i,j} = (x_{i,j,1}, x_{i,j,2}, \ldots, x_{i,j,n_{m}}) \), and the individual scalar components \( x_{i,j,k} \) (\( k = 1, 2, \ldots, n_{m} \)) of module \( x_{i,j} \) are called functional features. Each functional feature consists of several attributes \( a_{i,j,k,t} \) (\( t = 1, 2, \ldots, t_{n} \)), representing the function, \( x_{i,j,k} = (a_{i,j,k,1}, a_{i,j,k,2}, \ldots, a_{i,j,k,t_{n}}) \), where \( t_{n} \) is the number of attributes represented by TCO. We define five functional feature attributes: description, input energy, output energy, operand, and medium. Figure 4.9 shows the functional hierarchy for a drill as an example along with the levels for representing a product.
4.2.5 Case Study: Family of Power Tools

The Delta power tool family consists of six cordless power tools, each having a different functionality or use as shown in Figure 4.10. The Jigsaw has a blade attachment that is used to cut curves or different shapes. The Circular saw has a disc-like circular blade attachment that is used to saw through blocks and sheets. The Sander has a special base that is used to sand and smooth surfaces. The Drill has a drill bit that can be insulated and used to drill holes. The Brad Nailer has a magazine in which nails can be loaded and is used to drive nails through surfaces; its basic mechanism works like a gun. The Flashlight is used as a portable light.
The TCO for the Delta power tool family was developed using Protégé (Ver.3.2.1)\(^6\), a graphical editing tool that has functions for developing domain ontologies, customizing user interfaces, and integrating with other applications such as specific reasoning engines [60]. Based on TCO, we can determine the ontology properties: the classes, properties, and individuals to represent the six power tools.

The power tools were first divided into three classes - the function level, assembly level, and product specification level - and each class corresponds to a functional level, an assembly level, and a product specification, respectively. The function level consisted of two subclasses - behavior and function - and these classes have subclasses that are created for each product with unique properties. In the assembly level, the assembly subclass and the connection subclass represent the assembly information and the relation between components, respectively. These classes also have subclasses for each product.

\(^6\) http://protege.stanford.edu
with unique properties like a function module. The ‘product specification level’ class has subclasses for defining the specification of each product component. Figure 4.11 shows the power tool class and all its subclasses.

Figure 4.11: The Delta Power Tool Classes and Its Subclasses of ‘Function’

An instance in the subclass inherits all the properties of the parent class when subclasses or child classes are created. For example, the class ‘function’ is defined with properties such as ‘Subfunction’. All subclasses (i.e., child classes) inherit this property. For instance, the ‘Circular saw function’ subclass of ‘function’ also has ‘Subfunction’ as a property. As discussed in Section 4.2.4, for defining the matching information of related behavior, function, assembly, and connection, the property of each class can have a different class’s property like a property of the ‘assembly’ class. For example, the property of ‘function’ class has the ‘assembly information’ property representing the ‘assembly’ class.
Once the ontology based on TCO is completely implemented (refer to Appendix B), we can then develop a knowledgebase for finding the components that meet customer needs and determine the assembly relation and the other components related with customer needs. We used JARE\(^7\) to model the knowledgebase for the Delta power tool family. JARE is an environment for using logical inference in Java. Figure 4.12 shows an example of the knowledge representing the circular saw. From this knowledge, we know the specific component’s product specifications as well as basic information related to a query for functional requirements. Knowledge based on TCO for products and their components helps to promote component sharing as well as facilitate capturing various constraints and characteristics related with components and products.

\(^7\) http://jare.sourceforge.net

Figure 4.12: Examples of Rules and Facts for Design Knowledge

---

\(\text{Rules:}\
\{(\text{component} \ ?f \ \text{motor})\)\
\{(\text{function} \ ?f)\) (\text{input\_energy} \ ?f \ ?ie) (\text{output\_energy} \ ?f \ ?oe)\)\
\{(\text{operand} \ ?oe \ \text{force})\) (\text{medium} \ ?oe \ \text{shaft})\)\}

\{(\text{component} \ ?f \ \text{switch})\)\
\{(\text{function} \ ?f)\) (\text{input\_energy} \ ?f \ ?ie) (\text{output\_energy} \ ?f \ ?oe)\)\
\{(\text{operand} \ ?oe \ \text{electric\_energy})\) (\text{medium} \ ?oe \ \text{wire})\)\}

\(\text{Facts:}\
\{(\text{function} \ \text{convert})\)\
\{(\text{function} \ \text{import})\)\
\{(\text{input\_energy} \ \text{convert} \ \text{electric})\)\
\{(\text{input\_energy} \ \text{convert} \ \text{mechanic})\)\
\{(\text{output\_energy} \ \text{convert} \ \text{electric})\)\
\{(\text{output\_energy} \ \text{convert} \ \text{mechanic})\)\
\{(\text{operand} \ \text{mechanic} \ \text{force})\)\
\{(\text{operand} \ \text{electric} \ \text{electric\_energy})\)\
\{(\text{medium} \ \text{mechanic} \ \text{shaft})\)\
\{(\text{medium} \ \text{electric} \ \text{wire})\)\}
4.3 Ontology for Service Design

This section describes the process of developing an ontology for representing service design knowledge as shown in Figure 4.13. A service process model is introduced to describe a service based on a sequence using a graph model. For service analysis, a function-process matrix is used to identify the relationships between the service functions and the service processes that are offered as part of a service. Object-oriented concepts provide service analysis tools for describing a business process or a workflow process in a service. An ontology is applied to define properties that consist of attributes and behaviors for representing a service in a service hierarchy.

![Figure 4.13: The Process of Developing Service Ontology](image-url)
4.3.1 Service Analysis

Through service analysis, we can determine service-related design factors that are represented as processes, activities, objects, and/or features, as well as service functions and processes. These design factors are also used to define the properties of service components in service process model design. Based on service functions and processes, a function-process matrix (FPM) is introduced to identify the relationships between functional modules and process modules in a service. The FPM is similar to the function-component matrix (FCM) [84], which provides a mapping between a product’s components and its sub-functions. Table 4.2 shows a conceptual representation of the FPM based on checking account services. The first vertical column shows service functions, the top horizontal row is service processes, and the cells of the FPM represent the relationship between each function and process. The number ‘1’ in a cell indicates that a relationship among a function and a process exists. For example, Deposit in Table 4.2 consists of Make a Deposit and Certify ID to achieve its functional.

Table 4.2: A Function-Process Matrix for Service Analysis

<table>
<thead>
<tr>
<th>Functional module</th>
<th>Process module</th>
<th>Make a Deposit</th>
<th>Withdraw</th>
<th>Certify ID</th>
<th>Process module</th>
<th>Functional module</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deposit</td>
<td>Make a Deposit</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Withdraw</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Functional module</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Object-oriented concepts are used to support service analysis and representation combining ontologies. Table 4.3 provides the definitions of characteristics in object-oriented concepts [6, 28] and shows how to apply the concepts to represent services. For example, saving service processes can be represented by the characteristics of object-oriented concepts as show in Figure 4.14. In Figure 4.14, three objects for the saving service processes are identified as Customer, Employee, and Account. The objects can be described by their identity, behaviors, and states through behavioral structural modeling. The unified modeling language (UML) can be used to develop a behavior model and a structural model for services, since UML provides an excellent modeling setting [6].

![Figure 4.14: Saving Service Process and Applied to Object-Oriented Concepts](image-url)
In service design, the encapsulation concept provides methods for representing service components. Instantiation defines common features for creating a specific instance of a service based on the structure of a class. Figure 4.15 shows the encapsulation
concept and the instantiation relationship between a class and objects in service design and analysis. Specifically, the encapsulation concept is employed to develop a service process model in the proposed methodology.

![Image showing encapsulation and inheritance in service design](image)

**Figure 4.15: Extension Object-Oriented Concepts to Service Design**

The concept of inheritance can be applied to service module and component design for developing more robust and extensible services. As shown in Figure 4.16, we can identify the more specialized service components adding various features based on common features from a service module. The fundamental question regarding this is how to add these features. One has to build a domain specific ontology to capture relations and extensibility.
4.3.2 Service Process Model

A business process or workflow process is described by logically related activities to achieve a defined business goal or create value-added products or services to satisfy customer needs [69]. In services, a process can be considered as a procedure, routine, and/or policy to create services, which are defined by a set of activities, ordering constraints, and data or materials exchanged among the service activities. UML can be used to analyze service processes and/or basic workflow. UML is a standardized specification language for system design and analysis using a set of concepts, constructs, terminology, and notation [6]. For example, sequence diagrams are object-interaction diagrams that consider temporal sequencing and are useful for describing the behavior of use cases and the interaction between objects within a system [30]. Activity diagrams provide a modeling method to represent the business and operational workflows using the
detailed logic of a business rule. By analyzing the sequence diagrams or the activity diagrams for a service, we can obtain attributes and identify information flow among objects for service design. For instance, suppose that the objects of a deposit process in a banking service consist of a customer, an employee, a database, and a depository. A sequence diagram for the deposit process can be represented as shown in Figure 4.17. Processes in the diagram are represented by activities and attributes for performing the service.

![Sequence Diagram for a Banking Service](image)

**Figure 4.17: An Example of a Sequence Diagram for a Banking Service**

A process model can be defined by various languages with differences in their syntax and expressive rules [13]. In this research, a graph model is employed to describe a service process model based on service sequence diagrams. Graphs are an abstraction developed specifically to represent relationships and consist of two distinct parts: (1) nodes and (2) edges. The nodes are things in the graph that have relationships, and the edges are pairs of nodes connected by a relationship [8]. As shown in Figure 4.18, a node is defined as a service component (or an event) with properties that can describe service
processes, and an edge as direction presenting information, data, and materials flow. A node can be defined by five properties: (1) activity, (2) object, (3) input flow, (4) output flow, and (5) state. The activity is an event defining the intended interpretation of a service process between objects and is used as the name of a node. The object represents an object performing activities using input flow in certain services. The flow includes information, data, and materials, which occur in service processes. States are defined as things (objects) that change the input flow and the output flow. For example, a node changes the state of its inputs (states), such as information like a customer’s account balance or credit, materials like money, and data in a banking service.

![Service Process Model and Properties for a Node](image)

4.3.3 Service Representation

The basic idea of modular design is to organize services as a set of distinct service components that can be designed independently and develop a variety of services through the combination and standardization of service components. We assume that a service can be decomposed into modules that provide specific functions and processes, and
processes are through the combination of the modules’ attributes. To effectively define
the relationships between functional hierarchies in a service, an appropriate
representation scheme must be adopted for the services. In this work, a service ontology
is developed to represent the relationships between functional modules and process
modules as shown in Figure 4.16. In the service ontology, a process module has a
hierarchical structure to provide process representation-based semantics of services.

Suppose that a service family consists of \( l \) services, \( SF = (S_1, S_2, \ldots, S_l) \), and a
service consists of \( f \) functional modules, \( S_i = (y_{i,1}, y_{i,2}, \ldots, y_{i,f}, \ldots, y_{i,f}) \), where \( y_{i,f} \) denotes
service functional module \( f \) in service \( i \). For service processes, suppose that a service
consists of \( m \) service process modules, \( S_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,j}, \ldots, x_{i,m}) \), where \( x_{i,j} \) is process
module \( j \) in service \( i \) and consists of a vector of length \( n_m \), \( x_{i,j} = (x_{i,j,1}, x_{i,j,2}, \ldots, x_{i,j,k}, \ldots, x_{i,j,n_m}) \), and the individual scalar components \( x_{i,j,k} (k=1, 2, \ldots, n_m) \)
of a process module \( x_{i,j} \) are called \textit{process features}. Each process feature consists of
several attributes, \( a_{i,j,k,t} (t=1, 2, \ldots, t_n) \), representing the component,
\( x_{i,j,k} = (a_{i,j,k,1}, a_{i,j,k,2}, \ldots, a_{i,j,k,t}, \ldots, a_{i,j,k,t_n}) \), where \( t_n \) is the number of properties defined in
the service ontology. The identification of the properties is problem-dependent, but an
example can be found in the banking services case study in Section 4.3.4. Figure 4.19
shows the corresponding hierarchy for representing a family of checking account
services.
4.3.4 Case Study: Family of Checking Account Services

Consider a family of banking services consisting of four checking account services (see Table 4.5). The checking account services are designed for four different market segments based on customer’s preference, balance, credit, status, and so on. Using service analysis, we determine the service functions and service processes in this set of four services. A function-process matrix (FPM) was developed to identify relationship the service functions and processes in the checking account services as shown in Table 4.6.
### Table 4.4: Four Checking Account Services in a Banking Service Family

<table>
<thead>
<tr>
<th>Option</th>
<th>Service A</th>
<th>Service B</th>
<th>Service C</th>
<th>Service D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deposit</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Withdraw</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Transfer</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Banking Statement</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Online account statement</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Checking writing</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ATM transactions</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Online banking with bill pay</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Telephone banking</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Trade stocks online</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Optional business economic checking</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Maintenance fee</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Additional checking and saving account</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Loans and lines of credit</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Service for cashier’ check, and so on</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Interest</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Preferred rates on Money Market, CDs</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

### Table 4.5: The Function-Process Matrix for Four Checking Account Services

<table>
<thead>
<tr>
<th>Process module</th>
<th>Make a Deposit</th>
<th>Withdraw</th>
<th>Transfer Money</th>
<th>Trade Stocks</th>
<th>Check writing</th>
<th>Check Credit</th>
<th>Check Balance</th>
<th>Make a Loan</th>
<th>Open an Account</th>
<th>Operate Transactions</th>
<th>Record Transaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deposit</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Withdraw</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Transfer money</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Check writing</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ATM transactions</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Trade stocks online</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Additional checking and saving account</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Loans and lines of credit</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Service for cashier’ check, and so on</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Online banking with bill pay</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Interest</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Preferred rates on Money Market, CDs</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Telephone banking</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Optional Business Economy Checking</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Maintenance fee</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Based on the results of the service analysis, we can develop activity diagrams (see Figure 4.20) for service process modules to identify service processes or basic workflow as described in Section 4.3.2. Through these activity diagrams, we determine process features that are considered as the properties of the service components in the checking account services. A service process model for a service function was developed from service process modules and service process components. For example, Figure 4.21 shows a service process model for a deposit service function that consists of three service process modules: (1) Certify ID, (2) Make a Deposit, and (3) Record Transaction. The saving process module is composed of three components: request, accept, and inform. Each service process component has five properties as defined in Section 4.3.2.

Figure 4.20: Activity Diagram for a Deposit Process Function
The ontology for the four services was developed using Protégé. Figure 4.22 shows the checking account service classes and all subclasses in Protégé (refer to Appendix C).

Figure 4.21: Service Process Model for a Deposit Service Function

Figure 4.22: Checking Account Service Classes and Subclasses
4.4 Summary and Preview

In this chapter, the use of TCO for representing products and components is described based on ontologies to facilitate information reuse when developing new products. TCO development has been demonstrated using a case study involving six power tools. For service ontology development, a function-process matrix (FPM) was used to identify relationships between service functions and service processes in a family of services. Based on a graph model and a sequence diagram, a novel service process model was introduced to describe a service using ontologies and object-oriented concepts. The service ontology has been demonstrated to represent service design knowledge using a case study involving a family of four banking services. Because ontological representations have more semantics of products or services than ordinary representations using hierarchical data structures, customer needs and functional requirements defined by ontologies can help designers develop design knowledge for a collaborative and distributed environment. Ontological representations can provide necessary information to examine the feasibility of design alternatives and lead to improved knowledge discovery capabilities.

The next chapter examines module and platform identification for families of products and services using data mining techniques, namely, (1) clustering, (2) association rule mining, and (3) classification. Fuzzy set theory is used to create various platform strategies based on designers’ knowledge that are represented by linguistic design information. A design knowledge support system is introduced and implemented for discovering knowledge and supporting family design.
Chapter 5

Module and Platform Identification using Data Mining

The objective in this chapter is to develop a method for identifying a platform along with variant and unique modules in a product or service family using data mining techniques for Step 3 in the SMPDM. The chapter is organized as follows. Section 5.1 describes the proposed method for identifying a platform and modules using data mining. Section 5.2 presents a case study using a family of power tools. A case study for a service family is presented in Section 5.3. A summary of this chapter and a preview of the next are given in Section 5.4. Figure 5.1 shows the roadmap for this chapter.

Figure 5.1: Road Map for Chapter 5
5.1 A Method for Module and Platform Identification

To support Step 3 in the SMPDM, a novel method is introduced for supporting product and service family design using data mining techniques, namely, association rules, clustering, and classification. Techspecs Concepts Ontology (TCO) is used to represent the functions of a product as functional hierarchies as described in the previous Chapter. For services, service-based process analysis, ontologies, and object-oriented concepts are integrated to develop the method. The service ontology mentioned in Section 4.3 is used to represent the relationships between functions and processes in a service. Rules related to design knowledge are developed using association rule mining. Fuzzy c-means clustering is employed to partition functions into subsets for identifying a platform and modules in a given product family (or service) family. Based on the results of clustering and classification, fuzzy set theory is used to determine a product or service platform among common modules. Moreover, since design knowledge for products and services depends on the experience and knowledge of designers, representation of design knowledge, such as linguistic representation, may fail to enable a crisp description. When clustering design knowledge, we need to assign the knowledge to clusters with varying degrees of membership. Fuzzy membership can be used to represent and model the fuzziness of design knowledge [12].

Figure 5.2 shows a flow diagram of the proposed method that consists of three phases: (1) knowledge acquisition and representation, (2) module determination, and (3) platform and module identification. Phase 1 is related to Steps 1 and 2 in the SMPDM and is discussed in Chapters 3 and 4. Therefore, this chapter focuses on data mining
techniques related to Phases 2 and 3 in the proposed method by providing case studies. The next section discusses each phase of the method in detail.

### 5.1.1 Phase 1: Association Rule Mining for Knowledge Acquisition

An association rule describes an interesting relationship between attributes of different modules [1, 33]. Given a set of transactions, where each transaction is a set of attributes, an association rule is denoted as \( A \Rightarrow B \), where \( A \) and \( B \) are sets of attributes. More specifically, the association rule \( A \Rightarrow B \) indicates that transactions that contain attribute \( A \) tend to contain attribute \( B \). Support and confidence are used to assess the quality of the extracted rules [1]. The support of an attribute \( A \) in a set of transaction data
S means the probability of transaction data containing attribute A. The confidence of A \( \Rightarrow B \) represents the probability of attribute B occurring in S if attribute A occurs in S. An association rule with high confidence and support is called strong and is potentially useful for product design [1].

In association rule mining, transaction data is needed to develop rules related to product design. Based on TCO, we can develop transaction data that consists of several properties in the hierarchical functional relationship. For example, we can generate transaction data that is composed of module-level, functional-level, and attribute-level information in a product. In this research, the Apriori algorithm [3] is applied to generate association rules that use frequent item sets to define the association rules, since the Apriori algorithm uses databases consisting of transactions. Other algorithms such as *Partition*, *FP-growth*, and *Eclat* can be employed as desired [26]. A designer can extract important design guidelines from association rules. These design guidelines can be classified and translated into knowledge and rules for product design. Design rules can also be translated into specific integration and split rules that can be used to classify clusters. If two clusters have the same module, then the modules are combined by the integration rule. Otherwise, if a cluster has several modules, then the cluster is divided into the number of products in the cluster by the split rule (see Section 5.1.3) for more details.
5.1.2 Phase 2: Module Determination

**Fuzzy Clustering for Defining Modules**

Functional decomposition for a product is often represented in a hierarchical structure as discussed in Section 4.2.4. A hierarchical clustering method can classify a set of objects by measuring the similarity between objects [53]. Because heuristic methods for defining a module may provide overlapping or non-crisp boundaries among module clusters [83], the results of traditional clustering approaches are not appropriate to define clusters as modules in product design. Moreover, since design information for a product or a service depends on the experience and knowledge of designers, representation of design information, such as linguistic representation, may fail to enable a crisp description. When clustering design information, we need to assign the information to clusters with varying degrees of membership. Fuzzy membership can provide proper representation while also capturing the fuzziness of design knowledge [12]. Fuzzy clustering approaches can use fuzziness related to product design features and provide more useful solutions [47, 97]. In this research, we employ fuzzy c-means clustering (FCM) [10] to determine clusters for identifying modules for the product family. FCM is a clustering technique that is similar to k-means but uses fuzzy partitioning of data that is associated with different membership values between 0 and 1. Since FCM is an iterative algorithm, its aim is to find cluster centers that minimize a dissimilarity function. Let $X_k$ for $k = 1, 2, \ldots, n$ be a functional feature and a $d$-dimensional vector ($d$ is the number of attributes), and $u_{ik}$ the membership of $X_k$ to the $i$-th cluster ($i=1, 2, \ldots, c$). The $u_{ik}$ representing a fuzzy case is between 0 and 1. For example, if $u_{ik} = 0$, $u_{ik}$ has non-
membership to cluster $i$, and if $u_{ik} = 1$, then it has full membership. Values in between 0 and 1 indicate fractional membership.

Generally, FCM is defined as the solution of the following minimization problem [10]:

$$J_{FCM}(U, V) = \left\{ \sum_{i=1}^{c} \sum_{k=1}^{n} (u_{ik})^{m} \left\| X_{k} - v_{j} \right\|^{2} \right\}$$  \hspace{1cm} (5.1)

subject to:

$$\sum_{i=1}^{c} u_{ik} = 1 \quad \text{for all} \ k$$  \hspace{1cm} (5.2)

$$u_{ik} \in [0, 1]$$  \hspace{1cm} (5.3)

where $v_{j}$ is the cluster center of the $i$-th cluster that consists of a $d$-dimensional vector, and $m$ is a parameter ($m \geq 1$) that indicates the fuzziness of clusters. An algorithm for solving this problem is given in Refs. [10, 90]. This FCM algorithm does not ensure that it converges to a global optimal solution; however, it always converges to a local optimum that may lead to a different local minima according to different initial cluster centers [10, 90].

In this FCM algorithm, since the cluster number $c$ is determined before clustering, a validity index for an optimal $c$ should be considered for defining the number of clusters. In this research, the partition coefficient (PC) is used to determine the best cluster number $c$ [9]:

$$PC(c) = \frac{1}{n} \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^{2}$$  \hspace{1cm} (5.4)
where \(1/c \leq PC(c) \leq 1\). An optimal cluster number \(c^*\) maximizes \(PC(c)\), (the number of products +1) \(\leq c \leq n-1\).

The cluster number determines the number of modules. A maximum membership value in clusters is an indicator that represents the degree of the similarity among functional features in the clusters. Among clusters, clusters including the functional features for all selected products and services can be common modules for the platform.

In the proposed method, a coding approach is used to represent the attributes of product and service components for a given clustering method. The coding approach is problem-dependent, but an example can be found in the case studies in Sections 5.2 and 5.3.

*Classification using Integration and Split Rules*

Classification is a learning technique to separate distinct classes and map data into one of several predefined classes [34]. Classification approaches use rules that are usually developed from “learning” or “training” samples of pre-classified records [12, 34]. The process of generating classification rules is achieved using learning algorithms [12].

Since the clusters in FCM are determined based on functional features, additional analysis is needed to define the modules. Modules can be classified by integration rules and split rules extracted from association rules as mentioned in Section 5.1.1. For example, Figure 5.3 shows the classification process based on the integration and split rules. Clusters 1 and 2 have functional features \(x_{i,1,1}\) and \(x_{i,1,2}\), respectively. Cluster 3 has functional features \(x_{i,2,1}\) and \(x_{i,3,1}\). Since the functional features of Clusters 1 and 2 have
the same module features, they are integrated into module A. Meanwhile, Cluster 3 is split into two unique modules, B and C, since the functional features of Cluster 3 have different module features.

Figure 5.3: Process of Classifying Clusters

5.1.3 Phase 3: Module and Platform Identification

The clustering results provide membership values that represent the corresponding membership level of each cluster, which can be considered as the degree of similarity among functional features. Therefore, platform level can be determined as the membership value in a common. Based on fuzzy set theory [99], the membership values are measured using a rating scale of [0-1], and the ratings can be interpreted as fuzzy numbers based on different platform levels such as Low, Medium, and High. Let $X$ be a linguistic variable with the label “Platform level” with $U = [0, 1]$, and three fuzzy terms for the linguistic variable are defined as Low ($x_1$), Medium ($x_2$), and High ($x_3$) as shown in Figure 5.4. The membership function of each fuzzy set is assumed to be triangular, and the platform level can take three different linguistic terms. Platform level
membership functions are proposed to represent and determine the platform level of modules. Therefore, the membership values of functions in a common module are transferred into platform level values by the platform level membership functions. The platform level of the common module is determined by the maximum value among average membership level values for the module. For example, suppose two functional features are in a common module. If the membership values of the two functional features are 0.4 (A) and 0.6 (B), then the platform level values of the value 0.4 are represented by 0 at high, 0.8 at middle, and 0.2 at low, while the platform level values of the value 0.6 are represented by 0.2 at high, 0.8 at middle, and 0 at low. Therefore, the platform level of the common module is determined as the middle level (i.e., 0.1 at high, 0.8 at middle, and 0.1 at low).

![Fuzzy Membership Function Representing Platform Level](image)

**Figure 5.4: Fuzzy Membership Function Representing Platform Level**

The final results of the proposed method determine the platform along with the variant and unique modules for a product family where the platform consists of common modules with a high platform level that are shared. If variant modules are selected as a platform, additional functional features or costs will be required to make them a common
module. Since the classification based on design rules considers the hierarchical relationship among functional features, modules can be designed independently. In the conceptual stages of design, these results can help decision-makers by defining the set of modules for the product family. The effective set of modules will lead to improved product family design.

5.2 Case Study I: Power Tool Family

In this case study, five tools from the power tool family are investigated to demonstrate the proposed method (see Table 5.1), since the Flash Light is designed as a simple electronic mechanism and consists of fewer components compared to the other five tools. Currently, these products only have common modules related to electrical components. The proposed method determines if a more suitable platform and set of modules exists for the family. This case study focuses on a function-based platform for the power tool family during the conceptual design phase. The products representation for the five tools was developed using TCO (refer to Appendix B). Table 5.1 shows the 75 functional features of the selected five products. The five attributes of these functional features were coded using the values listed in Table 5.2. Each attribute takes a different code (number) based on functional features that are described as the functional basis proposed by Hirtz, et al. [27].
Table 5.1: Product Representation for Power Tool Family

<table>
<thead>
<tr>
<th>Product</th>
<th>Module</th>
<th>Functional features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circular saw</td>
<td>$x_{1,1}$</td>
<td>1(3, 5, 5, 15, 1), 2(9, 5, 5, 15, 1), 3(5, 5, 5, 15, 1)</td>
</tr>
<tr>
<td></td>
<td>$x_{1,2}$</td>
<td>1(13, 5, 5, 15, 1), 2(5, 9, 9, 2, 2)</td>
</tr>
<tr>
<td></td>
<td>$x_{1,3}$</td>
<td>1(4, 0, 5, 15, 10), 2(5, 5, 5, 15, 10), 3(14, 5, 5, 15, 10)</td>
</tr>
<tr>
<td></td>
<td>$x_{1,4}$</td>
<td>1(3, 1, 1, 2, 8), 2(14, 9, 9, 2, 8), 3(4, 9, 9, 2, 8)</td>
</tr>
<tr>
<td></td>
<td>$x_{1,5}$</td>
<td>1(3, 1, 1, 2, 7), 2(14, 1, 1, 2, 7)</td>
</tr>
<tr>
<td></td>
<td>$x_{1,6}$</td>
<td>1(3, 9, 9, 8, 2), 2(5, 9, 9, 8, 2)</td>
</tr>
<tr>
<td>Drill</td>
<td>$x_{2,1}$</td>
<td>1(3, 5, 5, 15, 1), 2(9, 5, 5, 15, 1), 3(5, 5, 5, 15, 1)</td>
</tr>
<tr>
<td></td>
<td>$x_{2,2}$</td>
<td>1(13, 5, 9, 2, 2), 2(5, 9, 9, 2, 2)</td>
</tr>
<tr>
<td></td>
<td>$x_{2,3}$</td>
<td>1(4, 0, 5, 15, 10), 2(5, 5, 5, 15, 10), 3(14, 5, 5, 15, 10)</td>
</tr>
<tr>
<td></td>
<td>$x_{2,4}$</td>
<td>1(3, 1, 1, 2, 4), 2(14, 9, 9, 2, 4), 3(4, 9, 9, 2, 4)</td>
</tr>
<tr>
<td></td>
<td>$x_{2,5}$</td>
<td>1(3, 1, 1, 2, 7), 2(14, 1, 1, 2, 7)</td>
</tr>
<tr>
<td></td>
<td>$x_{2,6}$</td>
<td>1(5, 9, 9, 8, 3), 2(11, 9, 9, 8, 3)</td>
</tr>
<tr>
<td>Jig saw</td>
<td>$x_{3,1}$</td>
<td>1(3, 5, 5, 15, 1), 2(9, 5, 5, 15, 1), 3(5, 5, 5, 15, 1)</td>
</tr>
<tr>
<td></td>
<td>$x_{3,2}$</td>
<td>1(13, 5, 9, 2, 2), 2(5, 9, 9, 2, 2)</td>
</tr>
<tr>
<td></td>
<td>$x_{3,3}$</td>
<td>1(13, 9, 9, 8, 3), 2(5, 9, 9, 8, 3)</td>
</tr>
<tr>
<td></td>
<td>$x_{3,4}$</td>
<td>1(4, 0, 5, 15, 10), 2(5, 5, 5, 15, 10), 3(14, 5, 5, 15, 10)</td>
</tr>
<tr>
<td></td>
<td>$x_{3,5}$</td>
<td>1(3, 1, 1, 2, 7), 2(14, 1, 1, 2, 7)</td>
</tr>
<tr>
<td></td>
<td>$x_{3,6}$</td>
<td>1(3, 1, 9, 2, 9), 2(14, 9, 9, 2, 9), 3(4, 9, 9, 2, 9)</td>
</tr>
<tr>
<td>Nailer</td>
<td>$x_{4,1}$</td>
<td>1(3, 5, 5, 15, 1), 2(9, 5, 5, 15, 1), 3(5, 5, 5, 15, 1)</td>
</tr>
<tr>
<td></td>
<td>$x_{4,2}$</td>
<td>1(13, 5, 9, 2, 2), 2(5, 9, 9, 2, 2)</td>
</tr>
<tr>
<td></td>
<td>$x_{4,3}$</td>
<td>1(13, 9, 9, 3, 11), 2(5, 9, 9, 3, 11)</td>
</tr>
<tr>
<td></td>
<td>$x_{4,4}$</td>
<td>1(4, 0, 5, 15, 10), 2(5, 5, 5, 15, 10), 3(14, 5, 5, 15, 10)</td>
</tr>
<tr>
<td></td>
<td>$x_{4,5}$</td>
<td>1(3, 1, 1, 2, 7), 2(14, 1, 1, 2, 7)</td>
</tr>
<tr>
<td></td>
<td>$x_{4,6}$</td>
<td>1(3, 1, 9, 2, 6), 2(4, 9, 9, 2, 6)</td>
</tr>
<tr>
<td>Sander</td>
<td>$x_{5,1}$</td>
<td>1(3, 5, 5, 15, 1), 2(9, 5, 5, 15, 1), 3(5, 5, 5, 15, 1)</td>
</tr>
<tr>
<td></td>
<td>$x_{5,2}$</td>
<td>1(13, 5, 9, 2, 2), 2(5, 9, 9, 2, 2)</td>
</tr>
<tr>
<td></td>
<td>$x_{5,3}$</td>
<td>1(13, 9, 9, 22, 12), 2(5, 9, 9, 22, 12)</td>
</tr>
<tr>
<td></td>
<td>$x_{5,4}$</td>
<td>1(4, 0, 5, 15, 10), 2(5, 5, 5, 15, 10), 3(14, 5, 5, 15, 10)</td>
</tr>
<tr>
<td></td>
<td>$x_{5,5}$</td>
<td>1(3, 1, 1, 2, 7), 2(14, 1, 1, 2, 7)</td>
</tr>
<tr>
<td></td>
<td>$x_{5,6}$</td>
<td>1(3, 1, 9, 2, 5), 2(14, 9, 9, 2, 5), 3(4, 9, 9, 2, 5)</td>
</tr>
</tbody>
</table>
5.2.1 Phase 1: Association Rule Mining for Knowledge Acquisition

Based on TCO, we develop transaction data consisting of module, function, and energy for each functional feature. For instance, Figure 5.5 shows the process of association rule mining based on the hierarchy functional relationship and transaction data for a circular saw. The Magnum Opus demo\textsuperscript{9} version 3.0 is used to generate association rules. The transaction data related to the five products is input as a text file to Magnum Opus (see Figure 5.6, which shows the input for and output from Magnum Opus). In the proposed method, we use support and strength among association metrics in Magnum Opus, because they provide support and confidence measures for the \textit{Apriori} algorithm, respectively. The generated rules are translated using the integration and split rules from Section 5.1.3 to classify modules. For example, if one cluster has two functional features and a rule with respect to the two features is not generated, then this cluster is identified as two modules by the split rule. Otherwise, if two clusters have two functional features and a rule with respect to these features is generated, then these clusters can be one module by the integration rule.

\textsuperscript{9} http://www.rulequest.com
Functional level

Electrical Module

Import
Actuate
Transfer

e.e.

Generate transaction data

Association rules

e.e. ⇒ Electrical, Support 0.20, Confidence 0.429
Battey ⇒ e.e., Support 0.20, Confidence 1.00

Rule mining algorithm

Figure 5.5: Process of Association Rule Mining

Figure 5.6: Input / Output of Magnum Opus for Power Tool Family
5.2.2 Phase 2: Fuzzy Clustering for Defining Modules

FCM was used next to determine modules for the five products. Since the number of clusters affects the number of initial modules, it is important to select the number of clusters for FCM effectively. An optimal cluster number \( c \) \((6 \leq c \leq 74)\) was estimated using the validity index (PC) mentioned in Section 5.1.2. As shown in Figure 5.7, the value of PC decreases as fuzziness increases. Figure 5.8 illustrates the values of PC for three different initial seeds at fuzziness 1.7 and 1,000 iterations. In this example, \( c^* = 13 \) was selected as the optimal cluster number to determine a platform and modules for the five products, since 12 to 15 clusters provide higher PC values than the other values. Table 5.3 shows the results of FCM using 13 clusters. Since Clusters 2, 4, 7, 8, 10, 12, and 13 have functional features for all five products, these clusters can be considered as common modules. Because FCM is very sensitive to initial parameter values, the clusters of the result should identify modules for obtaining a consistent solution.

![Figure 5.7: Values of PC for Fuzziness](image-url)
Figure 5.8: Values of the PC for Three Different Initial Seeds

Table 5.2: FCM Results (Membership Value) for 13 Clusters

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Circular saw</th>
<th>Drill</th>
<th>Jig saw</th>
<th>Nailer</th>
<th>Sander</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$x_{5,3,1}$ (0.87), $x_{5,3,2}$ (0.79)</td>
</tr>
<tr>
<td>2</td>
<td>$x_{1,2,2}$ (0.99), $x_{1,6,1}$ (0.33), $x_{1,6,2}$ (0.34)</td>
<td>$x_{2,2,2}$ (0.99), $x_{2,4,3}$ (0.67), $x_{2,6,1}$ (0.3)</td>
<td>$x_{3,2,2}$ (0.99), $x_{3,3,2}$ (0.3)</td>
<td>$x_{4,2,2}$ (0.99)</td>
<td>$x_{5,2,2}$ (0.99)</td>
</tr>
<tr>
<td>3</td>
<td>$x_{1,4,3}$ (0.48)</td>
<td>$x_{3,6,1}$ (0.23), $x_{3,6,3}$ (0.46)</td>
<td>$x_{4,3,2}$ (0.38), $x_{4,6,1}$ (0.22), $x_{4,6,3}$ (0.46)</td>
<td>$x_{5,6,1}$ (0.21), $x_{5,6,3}$ (0.36)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>$x_{1,3,1}$ (1)</td>
<td>$x_{2,3,1}$ (1)</td>
<td>$x_{3,4,1}$ (1)</td>
<td>$x_{4,4,1}$ (1)</td>
<td>$x_{5,4,1}$ (1)</td>
</tr>
<tr>
<td>5</td>
<td>$x_{1,4,2}$ (0.71)</td>
<td>$x_{3,6,2}$ (0.97)</td>
<td>$x_{4,3,1}$ (0.92)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$x_{5,4,2}$ (0.74)</td>
</tr>
<tr>
<td>7</td>
<td>$x_{1,4,1}$ (0.99), $x_{1,5,1}$ (1)</td>
<td>$x_{2,4,1}$ (0.89), $x_{2,5,1}$ (1)</td>
<td>$x_{3,5,1}$ (1)</td>
<td>$x_{4,5,1}$ (1)</td>
<td>$x_{5,5,1}$ (1)</td>
</tr>
<tr>
<td>8</td>
<td>$x_{1,3,1}$ (0.92), $x_{1,3,2}$ (0.89)</td>
<td>$x_{2,3,1}$ (0.92), $x_{2,3,2}$ (0.89)</td>
<td>$x_{3,4,1}$ (0.92), $x_{3,4,2}$ (0.89)</td>
<td>$x_{4,4,1}$ (0.92), $x_{4,4,2}$ (0.89)</td>
<td>$x_{5,4,1}$ (0.92), $x_{5,4,2}$ (0.89)</td>
</tr>
<tr>
<td>9</td>
<td>$x_{2,4,2}$ (0.85), $x_{2,6,2}$ (0.24)</td>
<td>$x_{3,3,1}$ (0.92)</td>
<td></td>
<td></td>
<td>$x_{5,6,2}$ (0.57)</td>
</tr>
<tr>
<td>10</td>
<td>$x_{1,1,1}$ (0.95), $x_{1,1,2}$ (0.77), $x_{1,1,3}$ (1)</td>
<td>$x_{2,1,1}$ (0.95), $x_{2,1,2}$ (0.77), $x_{2,1,3}$ (1)</td>
<td>$x_{3,1,1}$ (0.95), $x_{3,1,2}$ (0.77), $x_{3,1,3}$ (1)</td>
<td>$x_{4,1,1}$ (0.95), $x_{4,1,2}$ (0.77), $x_{4,1,3}$ (1)</td>
<td>$x_{5,1,1}$ (0.95), $x_{5,1,2}$ (0.77), $x_{5,1,3}$ (1)</td>
</tr>
<tr>
<td>11</td>
<td>$x_{1,4,3}$ (0.48)</td>
<td>$x_{3,6,1}$ (0.23), $x_{3,6,3}$ (0.46)</td>
<td>$x_{4,3,2}$ (0.38), $x_{4,6,1}$ (0.22), $x_{4,6,3}$ (0.46)</td>
<td>$x_{5,6,1}$ (0.21), $x_{5,6,3}$ (0.36)</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>$x_{1,5,2}$ (1)</td>
<td>$x_{2,5,2}$ (1)</td>
<td>$x_{3,5,2}$ (1)</td>
<td>$x_{4,5,2}$ (1)</td>
<td>$x_{5,5,2}$ (1)</td>
</tr>
<tr>
<td>13</td>
<td>$x_{1,2,1}$ (1)</td>
<td>$x_{2,2,1}$ (1)</td>
<td>$x_{3,2,1}$ (1)</td>
<td>$x_{4,2,1}$ (1)</td>
<td>$x_{5,2,1}$ (1)</td>
</tr>
</tbody>
</table>
Based on the integration and split rules, 13 clusters were identified as modules. For example, since two functional features \((x_{3,2,2}, x_{3,3,2})\) of the jig saw in Cluster 2 are related to different modules—a motor module and a conversion module—the features of the jig saw were decomposed into two different modules. Figure 5.9 shows the split rule for the generated rules of the jig saw. The features of the circular saw and the drill in Cluster 2 were decomposed into different modules using the split rule. Cluster 4 and Cluster 8 can be combined into one module using the integration rule, since these features in each product have been obtained from the same module and rules related to the functional features and the modules were generated. For instance, in the Clusters 4 and 8, the three functional features \((x_{1,3,1}, x_{1,2,2}, x_{1,3,3})\) of the circular saw are integrated as shown in Figure 5.9.

![Figure 5.9: Examples of Integration Rule and Split Rule](image-url)
5.2.3 Phase 3: Module and Platform Identification

Using the platform level membership function described in Phase 3, the platform levels of the clusters were determined as shown in Table 5.3. The existing common modules in the actual product family shown in Table 5.4 are a battery module and a battery terminal connector in an input module. Based on the high platform level for the five power tools, a proposed platform consists of functions related to an electronic module, a motor module, a battery module, and an input module. Finally, one platform consisting of 4 common modules and 19 variant and unique modules are identified for the product family. Comparing this to the existing platform for the five products, the number of common modules can be increased by 23% based on common functional features. During conceptual design, this information can provide designers with guidelines for effective product family development. In terms of the existing products, the proposed method can give a metric to assess commonality between products for functional features.
Table 5.3: New Platform and Modules for Five Products (Platform Level Value)

<table>
<thead>
<tr>
<th>Design</th>
<th>Circular saw</th>
<th>Drill</th>
<th>Jig saw</th>
<th>Nailer</th>
<th>Sander</th>
</tr>
</thead>
<tbody>
<tr>
<td>Platform</td>
<td>[x_{1,1,1}, x_{1,1,2}, x_{1,1,3}], [x_{2,1,1}, x_{2,1,2}, x_{2,1,3}], [x_{3,1,1}, x_{3,1,2}, x_{3,1,3}], [x_{4,1,1}, x_{4,1,2}, x_{4,1,3}], [x_{5,1,1}, x_{5,1,2}, x_{5,1,3}], (0.813) - electronic module</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>[x_{1,3,1}, x_{1,3,2}, x_{1,3,3}], [x_{2,3,1}, x_{2,3,2}, x_{2,3,3}], [x_{3,3,1}, x_{3,3,2}, x_{3,3,3}], [x_{4,3,1}, x_{4,3,2}, x_{4,3,3}], (0.98) - battery module</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>[x_{1,5,1}, x_{1,5,2}], [x_{2,5,1}, x_{2,5,2}], [x_{3,5,1}, x_{3,5,2}], [x_{4,5,1}, x_{4,5,2}], [x_{5,5,1}, x_{5,5,2}], (0.99) - motor module</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Module</td>
<td>x_{1,6,1}, x_{1,6,2}, x_{2,4,3}, x_{2,6,1}, x_{3,3,2}, x_{3,5,1}, x_{3,5,2}</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>[x_{1,4,1}, x_{1,4,2}, x_{1,4,3}, x_{2,4,1}, x_{2,4,2}, x_{2,4,3}, x_{3,5,1}, x_{3,5,2}]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5.4: Comparison of Existing Products and Proposed Products

<table>
<thead>
<tr>
<th>Existing</th>
<th>Proposed</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Common</strong></td>
<td>• Battery module (x_{1,3}*)&lt;br&gt;• Battery terminal connector (x_{1,1})</td>
</tr>
<tr>
<td></td>
<td>• Electronic module&lt;br&gt;• Battery module&lt;br&gt;• Motor module&lt;br&gt;• Input module</td>
</tr>
<tr>
<td><strong>Variant or Unique</strong></td>
<td>• Electronic module (x_{1,4}<em>)&lt;br&gt;• Motor module (x_{2,4}</em>)&lt;br&gt;• Input module (x_{3,4}<em>)&lt;br&gt;• Gear train module (x_{4,4}</em>)&lt;br&gt;• Blade module (x_{5,4})<em>&lt;br&gt;• Bit module (x_{6,4}</em>)&lt;br&gt;• Conversion module (x_{5,3}<em>, x_{5,5}</em>)&lt;br&gt;• Nail hitter module (x_{4,3}<em>)&lt;br&gt;• Magazine module (x_{4,6}</em>)&lt;br&gt;• Sander mounting module (x_{5,6}*)</td>
</tr>
<tr>
<td></td>
<td>• Gear train module&lt;br&gt;• Conversion module</td>
</tr>
<tr>
<td><strong>Unique</strong></td>
<td>• Blade module&lt;br&gt;• Sander mounting module&lt;br&gt;• Magazine module&lt;br&gt;• Bit module (x_{2,4}*)&lt;br&gt;• Gear train module (x_{2,6,2})&lt;br&gt;• Conversion module (x_{3,1})&lt;br&gt;• Sander mounting module (x_{5,6,2})</td>
</tr>
</tbody>
</table>
5.3 Case Study II: Checking Account Services

Next we consider the four checking account services introduced in Section 4.3.4. Using the proposed method, we determine a platform and a set of modules for the banking service family. This case study focuses on a process-based platform for the family of banking services at the conceptual stage of development.

5.3.1 Phase 1: Knowledge Representation

The ontology for the four services was developed in Section 4.3.4 (refer to Appendix C). The five attributes of these process features were coded using the values listed in Table 5.5. Each property takes a different code (number). Process features in Table 5.5 are developed based on the results of the service process analysis for the four checking account services. For instance, if the properties of a node consist of accept (activity), employee (object), money (input flow), amount (output flow), and balance (operand), then the codes for the properties are 1, 2, 4, 6, and 2, respectively. Table 5.6 shows the 103 process features of the selected four services. The properties of these process features are coded using the numbers in Table 5.5.
### Table 5.5: Property Codes for Process Features in the Four Checking Account Services

<table>
<thead>
<tr>
<th>Code</th>
<th>Activity</th>
<th>Object</th>
<th>Flow (contents)</th>
<th>Operand</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Accept</td>
<td>Customer</td>
<td>Customer ID</td>
<td>Credit</td>
</tr>
<tr>
<td>2</td>
<td>Confirm</td>
<td>Employee</td>
<td>Account No.</td>
<td>Balance</td>
</tr>
<tr>
<td>3</td>
<td>Inform</td>
<td>Database</td>
<td>Credit</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Query</td>
<td>Trading (Employee)</td>
<td>Money</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Request</td>
<td>Depository</td>
<td>Employee ID</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Reject</td>
<td></td>
<td>Amount</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Proposal</td>
<td></td>
<td>Balance</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td>Message</td>
<td></td>
</tr>
</tbody>
</table>
Table 5.6: Service Representation for Four Checking Account Services

<table>
<thead>
<tr>
<th>Service</th>
<th>Module Process</th>
<th>Component</th>
<th>Attributes</th>
<th>Attribute codes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Activity</td>
<td>Object</td>
<td>Input flow</td>
<td>Output flow</td>
</tr>
<tr>
<td></td>
<td>Activity</td>
<td>Object</td>
<td>Input flow</td>
<td>Output flow</td>
</tr>
<tr>
<td></td>
<td>Activity</td>
<td>Object</td>
<td>Input flow</td>
<td>Output flow</td>
</tr>
<tr>
<td></td>
<td>Activity</td>
<td>Object</td>
<td>Input flow</td>
<td>Output flow</td>
</tr>
<tr>
<td>Make a Deposit</td>
<td>X3,11,1</td>
<td>Request</td>
<td>Customer No.</td>
<td>Amount Balance</td>
</tr>
<tr>
<td></td>
<td>X3,11,2</td>
<td>Accept</td>
<td>Employee Money</td>
<td>Amount Balance</td>
</tr>
<tr>
<td></td>
<td>X3,11,3</td>
<td>Inform</td>
<td>Database Amount</td>
<td>Amount Balance</td>
</tr>
<tr>
<td>Withdraw</td>
<td>X2,9,1</td>
<td>Request</td>
<td>Customer No.</td>
<td>Amount Balance</td>
</tr>
<tr>
<td></td>
<td>X2,9,2</td>
<td>Accept</td>
<td>Employee Money</td>
<td>Amount Balance</td>
</tr>
<tr>
<td></td>
<td>X2,9,3</td>
<td>Inform</td>
<td>Database Amount</td>
<td>Amount Balance</td>
</tr>
<tr>
<td>Transfer Money</td>
<td>X4,8,1</td>
<td>Query</td>
<td>Employee Amount</td>
<td>Amount Balance</td>
</tr>
<tr>
<td></td>
<td>X4,8,2</td>
<td>Accept</td>
<td>Employee Money</td>
<td>Amount Balance</td>
</tr>
<tr>
<td></td>
<td>X4,8,3</td>
<td>Inform</td>
<td>Database Amount</td>
<td>Amount Balance</td>
</tr>
<tr>
<td>Trade Stocks</td>
<td>X3,7,1</td>
<td>Query</td>
<td>Employee Account No.</td>
<td>Message</td>
</tr>
<tr>
<td></td>
<td>X3,7,2</td>
<td>Accept</td>
<td>Employee Message</td>
<td>Message</td>
</tr>
<tr>
<td></td>
<td>X3,7,3</td>
<td>Inform</td>
<td>Database Message</td>
<td>Message</td>
</tr>
<tr>
<td>Check Credit</td>
<td>X2,6,1</td>
<td>Query</td>
<td>Employee</td>
<td>Message</td>
</tr>
<tr>
<td></td>
<td>X2,6,2</td>
<td>Accept</td>
<td>Employee Message</td>
<td>Message</td>
</tr>
<tr>
<td></td>
<td>X2,6,3</td>
<td>Inform</td>
<td>Database Message</td>
<td>Message</td>
</tr>
<tr>
<td>Check Balance</td>
<td>X1,5,1</td>
<td>Query</td>
<td>Employee Account No.</td>
<td>Message</td>
</tr>
<tr>
<td></td>
<td>X1,5,2</td>
<td>Accept</td>
<td>Employee Message</td>
<td>Message</td>
</tr>
<tr>
<td></td>
<td>X1,5,3</td>
<td>Inform</td>
<td>Database Message</td>
<td>Message</td>
</tr>
<tr>
<td>Record Transaction</td>
<td>X4,7,1</td>
<td>Confirm</td>
<td>Database Account No.</td>
<td>Amount Balance</td>
</tr>
<tr>
<td></td>
<td>X4,7,2</td>
<td>Accept</td>
<td>Employee Amount</td>
<td>Amount Balance</td>
</tr>
<tr>
<td></td>
<td>X4,7,3</td>
<td>Inform</td>
<td>Database Amount</td>
<td>Amount Balance</td>
</tr>
</tbody>
</table>

**Service A**

- **Make a Deposit**
  - X3,11,1: Request Customer No. Amount Balance
  - X3,11,2: Accept Employee Money Amount Balance
  - X3,11,3: Inform Database Amount Amount Balance

- **Withdraw**
  - X2,9,1: Request Customer No. Amount Balance
  - X2,9,2: Accept Employee Money Amount Balance
  - X2,9,3: Inform Database Amount Amount Balance

- **Transfer Money**
  - X4,8,1: Query Employee Amount Amount Balance
  - X4,8,2: Accept Employee Money Amount Balance
  - X4,8,3: Inform Database Amount Amount Balance

- **Trade Stocks**
  - X3,7,1: Query Employee Account No. Message
  - X3,7,2: Accept Employee Message Message
  - X3,7,3: Inform Database Message Message

- **Check Credit**
  - X2,6,1: Query Employee Message Message
  - X2,6,2: Accept Employee Message Message
  - X2,6,3: Inform Database Message Message

- **Check Balance**
  - X1,5,1: Query Employee Account No. Message
  - X1,5,2: Accept Employee Message Message
  - X1,5,3: Inform Database Message Message

- **Record Transaction**
  - X4,7,1: Confirm Database Account No. Amount Balance
  - X4,7,2: Accept Employee Amount Amount Balance
  - X4,7,3: Inform Database Amount Amount Balance

**Service B**

- **Make a Deposit**
  - X3,11,1: Request Customer No. Amount Balance
  - X3,11,2: Accept Employee Money Amount Balance
  - X3,11,3: Inform Database Amount Amount Balance

- **Withdraw**
  - X2,9,1: Request Customer No. Amount Balance
  - X2,9,2: Accept Employee Money Amount Balance
  - X2,9,3: Inform Database Amount Amount Balance

- **Transfer Money**
  - X4,8,1: Query Employee Amount Amount Balance
  - X4,8,2: Accept Employee Money Amount Balance
  - X4,8,3: Inform Database Amount Amount Balance

- **Trade Stocks**
  - X3,7,1: Query Employee Account No. Message
  - X3,7,2: Accept Employee Message Message
  - X3,7,3: Inform Database Message Message

- **Check Credit**
  - X2,6,1: Query Employee Message Message
  - X2,6,2: Accept Employee Message Message
  - X2,6,3: Inform Database Message Message

- **Check Balance**
  - X1,5,1: Query Employee Account No. Message
  - X1,5,2: Accept Employee Message Message
  - X1,5,3: Inform Database Message Message

- **Record Transaction**
  - X4,7,1: Confirm Database Account No. Amount Balance
  - X4,7,2: Accept Employee Amount Amount Balance
  - X4,7,3: Inform Database Amount Amount Balance

**Service C**

- **Make a Deposit**
  - X3,11,1: Request Customer No. Amount Balance
  - X3,11,2: Accept Employee Money Amount Balance
  - X3,11,3: Inform Database Amount Amount Balance

- **Withdraw**
  - X2,9,1: Request Customer No. Amount Balance
  - X2,9,2: Accept Employee Money Amount Balance
  - X2,9,3: Inform Database Amount Amount Balance

- **Transfer Money**
  - X4,8,1: Query Employee Amount Amount Balance
  - X4,8,2: Accept Employee Money Amount Balance
  - X4,8,3: Inform Database Amount Amount Balance

- **Trade Stocks**
  - X3,7,1: Query Employee Account No. Message
  - X3,7,2: Accept Employee Message Message
  - X3,7,3: Inform Database Message Message

- **Check Credit**
  - X2,6,1: Query Employee Message Message
  - X2,6,2: Accept Employee Message Message
  - X2,6,3: Inform Database Message Message

- **Check Balance**
  - X1,5,1: Query Employee Account No. Message
  - X1,5,2: Accept Employee Message Message
  - X1,5,3: Inform Database Message Message

- **Record Transaction**
  - X4,7,1: Confirm Database Account No. Amount Balance
  - X4,7,2: Accept Employee Amount Amount Balance
  - X4,7,3: Inform Database Amount Amount Balance

**Service D**

- **Make a Deposit**
  - X3,11,1: Request Customer No. Amount Balance
  - X3,11,2: Accept Employee Money Amount Balance
  - X3,11,3: Inform Database Amount Amount Balance

- **Withdraw**
  - X2,9,1: Request Customer No. Amount Balance
  - X2,9,2: Accept Employee Money Amount Balance
  - X2,9,3: Inform Database Amount Amount Balance

- **Transfer Money**
  - X4,8,1: Query Employee Amount Amount Balance
  - X4,8,2: Accept Employee Money Amount Balance
  - X4,8,3: Inform Database Amount Amount Balance

- **Trade Stocks**
  - X3,7,1: Query Employee Account No. Message
  - X3,7,2: Accept Employee Message Message
  - X3,7,3: Inform Database Message Message

- **Check Credit**
  - X2,6,1: Query Employee Message Message
  - X2,6,2: Accept Employee Message Message
  - X2,6,3: Inform Database Message Message

- **Check Balance**
  - X1,5,1: Query Employee Account No. Message
  - X1,5,2: Accept Employee Message Message
  - X1,5,3: Inform Database Message Message

- **Record Transaction**
  - X4,7,1: Confirm Database Account No. Amount Balance
  - X4,7,2: Accept Employee Amount Amount Balance
  - X4,7,3: Inform Database Amount Amount Balance
5.3.2 Phase 2: Fuzzy Clustering for Defining Modules

FCM was used to determine modules for the four checking account services. An optimal cluster number $c \ (5 \leq c \leq 102)$ was estimated using the validity index (PC) defined as in Section 5.1.2. Figure 5.10 illustrates the values of PC for three different initial seeds at fuzziness $= 1.7$ and for 10,000 iterations. In this example, $c = 11$ was selected as the cluster number to determine a platform and modules for the four services, since 10 to 15 clusters provides higher average PC values than the other values. Table 5.7 shows the results of FCM using 11 clusters. Clusters that have process features for all four services can be considered as common modules.

![Figure 5.10: Values of the PC for Three Different Initial Seeds](image-url)
## 5.3.3 Phase 3: Modules and Platform Identification

Using the platform level membership function described in Section 5.1.3, the platform levels of the clusters were determined as shown in Table 5.8. Since the platform level values of Clusters 1, 3, 4, 7, and 8 indicate high platform level, these common modules can be combined into the platform for this family of four banking services.

### Table 5.7: Clustering Results for the Four Checking Account Services

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Service A</th>
<th>Service B</th>
<th>Service C</th>
<th>Service D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$X_{6,1}$ $X_{6,2}$</td>
<td>$X_{6,1}$ $X_{6,2}$</td>
<td>$X_{6,1}$ $X_{6,2}$</td>
<td>$X_{6,1}$ $X_{6,2}$</td>
</tr>
<tr>
<td>2</td>
<td>$X_{6,3}$ $X_{6,4}$</td>
<td>$X_{6,3}$ $X_{6,4}$</td>
<td>$X_{6,3}$ $X_{6,4}$</td>
<td>$X_{6,3}$ $X_{6,4}$</td>
</tr>
<tr>
<td>3</td>
<td>$X_{6,5}$ $X_{6,6}$</td>
<td>$X_{6,5}$ $X_{6,6}$</td>
<td>$X_{6,5}$ $X_{6,6}$</td>
<td>$X_{6,5}$ $X_{6,6}$</td>
</tr>
<tr>
<td>4</td>
<td>$X_{6,7}$ $X_{6,8}$</td>
<td>$X_{6,7}$ $X_{6,8}$</td>
<td>$X_{6,7}$ $X_{6,8}$</td>
<td>$X_{6,7}$ $X_{6,8}$</td>
</tr>
<tr>
<td>5</td>
<td>$X_{6,9}$ $X_{6,10}$</td>
<td>$X_{6,9}$ $X_{6,10}$</td>
<td>$X_{6,9}$ $X_{6,10}$</td>
<td>$X_{6,9}$ $X_{6,10}$</td>
</tr>
<tr>
<td>6</td>
<td>$X_{6,11}$ $X_{6,12}$</td>
<td>$X_{6,11}$ $X_{6,12}$</td>
<td>$X_{6,11}$ $X_{6,12}$</td>
<td>$X_{6,11}$ $X_{6,12}$</td>
</tr>
<tr>
<td>7</td>
<td>$X_{6,13}$ $X_{6,14}$</td>
<td>$X_{6,13}$ $X_{6,14}$</td>
<td>$X_{6,13}$ $X_{6,14}$</td>
<td>$X_{6,13}$ $X_{6,14}$</td>
</tr>
<tr>
<td>8</td>
<td>$X_{6,15}$ $X_{6,16}$</td>
<td>$X_{6,15}$ $X_{6,16}$</td>
<td>$X_{6,15}$ $X_{6,16}$</td>
<td>$X_{6,15}$ $X_{6,16}$</td>
</tr>
<tr>
<td>9</td>
<td>$X_{6,17}$ $X_{6,18}$</td>
<td>$X_{6,17}$ $X_{6,18}$</td>
<td>$X_{6,17}$ $X_{6,18}$</td>
<td>$X_{6,17}$ $X_{6,18}$</td>
</tr>
<tr>
<td>10</td>
<td>$X_{6,19}$ $X_{6,20}$</td>
<td>$X_{6,19}$ $X_{6,20}$</td>
<td>$X_{6,19}$ $X_{6,20}$</td>
<td>$X_{6,19}$ $X_{6,20}$</td>
</tr>
<tr>
<td>11</td>
<td>$X_{6,21}$ $X_{6,22}$</td>
<td>$X_{6,21}$ $X_{6,22}$</td>
<td>$X_{6,21}$ $X_{6,22}$</td>
<td>$X_{6,21}$ $X_{6,22}$</td>
</tr>
</tbody>
</table>
The clusters for the suggested service platform embody a Request module, a Query module, an Accept module, and an Inform module in terms of the activities listed in Table 5.7. Therefore, the platform for the checking account services can be designed by integrating processes that are related to these activities involving a customer and an employee. Variant and unique modules can be used to increase the number of services according to customers’ needs or functional requirements. The service ontology can help a designer search appropriate process features related to particular service functions and processes for service design. During the conceptual stages of development, this information can provide designers with guidelines for effective service family design.

<table>
<thead>
<tr>
<th>cluster</th>
<th>Platform level</th>
<th>Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>low</td>
<td>middle</td>
</tr>
<tr>
<td>3</td>
<td>0.0039</td>
<td>0.1118</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0.0114</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0.1088</td>
<td>0.0932</td>
</tr>
<tr>
<td>2</td>
<td>0.0755</td>
<td>0.3711</td>
</tr>
<tr>
<td>5/9</td>
<td>0.0903</td>
<td>0.9097</td>
</tr>
<tr>
<td>6</td>
<td>0.0339</td>
<td>0.4048</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0.3886</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>0.2257</td>
</tr>
</tbody>
</table>
5.4 Summery and Preview

In this chapter, a new method is proposed for identifying a module-based platform along with variant and unique modules in product and service families using data mining techniques. Fuzzy c-means clustering was employed to cluster the functional features of products and services based on the similarity among them. The resulting clusters yield the initial modules, which are then identified through classification. Fuzzy set theory was used to identify the platform within the family. Case studies involving a product family and a service family have demonstrated to work the proposed method for determining a new platform.

The next chapter provides agent-based decision-making for strategic platform design for product and service families in a dynamic and uncertain market environment. The architecture of a multi-agent system (MAS) is proposed along with specific agents’ roles and knowledge. A market-based negotiation mechanism is introduced to determine a platform in dynamic market environment and a game theoretic approach is used to develop a strategic platform design game with incomplete information.
Chapter 6

Agent-based Decision-Making for Determining a Platform Strategy

As Step 4 in the SMPDM, this chapter introduces agent-based decision-making approaches to determine a platform design strategy for a product or service family in dynamic and uncertain market environments. The results of Step 3 provide information for developing platform design strategies that can be used for agents’ knowledge in the proposed multi-agent system. Section 6.1 describes a mathematical programming model for modular platform design. A dynamic multi-agent system (DMAS) based on an electronic market is proposed to support agent-based decision-making in Section 6.2. Section 6.3 describes a market-based negotiation mechanism for the DMAS and a case study to show how the DMAS can be used to design a product platform using negotiation mechanisms. A game theoretic approach is used to determine an optimal platform strategy for a product family in an uncertain market environment in Section 6.4. In Section 6.5, the concept of the strategic platform design for a product family is extended to service family design based on the game theoretic approach. Section 6.6 provides a summary of this chapter and a preview of the next. Figure 6.1 shows the road map for this chapter.
6.1 Module-based Platform and Mathematical Model for a Product Family

In this research, we introduce an engineering parameter (EP) module based on module-based design approaches for developing module-based platform strategies by the trade-off of relationships between unique and common modules. The next section discusses EP modules in detail.

6.1.1 Engineering Parameter Cost Model for Product Family

Modules can be categorized based on function into: (1) unique, (2) common, and (3) variant or engineering parameter (EP) modules. Unique modules are based on
distinctive functions within a product family - components in these modules cannot be replaced by those in different modules to fulfill their task. Different options within the product family can be designed as unique modules to create a variety of products. Common modules are based on common functions within a product family so that components in the modules can be shared. Variant or EP modules are based on common functions but differ in having different EP values. An EP module is a combination of one or several components that vary between products based on its EP values. The EP of a component is a representative engineering parameter that is used for component selection in the market, and each EP component can be provided by a number of suppliers. For example, a motor in an EP module is represented by several EP values (e.g., torque, speed). If EP values that are selected for a platform do not meet the EP values of corresponding EP modules across a product family, additional EP components (i.e., gears with high ratios) are required to improve the EP values in terms of their functional requirements. As such, EP costs include the costs caused by changing EP components to increase the functionality of common EP modules.

Let $EP_{ik}$ be the EP value of EP module $i$ in product $k$; let $EPC_{ik}$ be the EP component costs at $EP_{ik}$; let $AC_{ik}(\alpha)$ the assembly cost of EP module $i$ in product $k$. The EP costs for EP module $i$ in product $k$ can be formulated at the module level as follows:

$$\text{EP costs} = \mu EPC_{ik} + D_{(ik,j)} + AC_{ik}(\alpha)$$

(6.1)

where $\alpha$ is a degree of cost reduction by commonality in assembly, $D_{(ik,j)}$ is the EP loss cost in product $j$ by sharing EP module $i$ in product $k$, and $\mu$ is discount ratio for sharing the number of EP modules. For example, if a module is unique, then its EP value is
calculated by combining the component cost and the assembly cost. If $EP_{ik}$ is selected for a platform, then the $EPC_{ik}$ is reduced to $\mu EPC_{ik}$ . Meanwhile $D_{(ik,j)}$ is incurred when $EP_{ik}$ does not meet $EP_{j}$. The EP loss cost can be determined by the additional component cost needed to satisfy the functional requirements. Assembly cost depends on the number of EP components. Commonality reduces the number of different EP components to be assembled, resulting in cost reduction ($AC_{ik} (\alpha)$). Table 6.1 represents these relationships in a matrix form for three EP (torque) modules of the family: $EP_{11}$ (5N.m.), $EP_{12}$ (7N.m.), and $EP_{13}$ (10N.m.).

<table>
<thead>
<tr>
<th>EP</th>
<th>$EP_{11}$ (5N.m.)</th>
<th>$EP_{12}$ (7N.m.)</th>
<th>$EP_{13}$ (10N.m.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unique</td>
<td>$EPC_{i1} + AC_{i1} (\alpha)$</td>
<td>$EPC_{i2} + AC_{i2} (\alpha)$</td>
<td>$EPC_{i3} + AC_{i3} (\alpha)$</td>
</tr>
<tr>
<td>Common</td>
<td>$\mu EPC_{i1} + AC_{i1} (\alpha)$</td>
<td>$\mu EPC_{i1} + AC_{i1} (\alpha) + D_{(i1,2)}$</td>
<td>$\mu EPC_{i1} + AC_{i1} (\alpha) + D_{(i1,3)}$</td>
</tr>
<tr>
<td>$EP_{11}$</td>
<td>$\mu EPC_{i2} + AC_{i2} (\alpha)$</td>
<td>$\mu EPC_{i2} + AC_{i2} (\alpha)$</td>
<td>$\mu EPC_{i2} + AC_{i2} (\alpha) + D_{(i2,3)}$</td>
</tr>
<tr>
<td>$EP_{12}$</td>
<td>$\mu EPC_{i3} + AC_{i3} (\alpha)$</td>
<td>$\mu EPC_{i3} + AC_{i3} (\alpha)$</td>
<td>$\mu EPC_{i3} + AC_{i3} (\alpha)$</td>
</tr>
<tr>
<td>$EP_{13}$</td>
<td>$\mu EPC_{i3} + AC_{i3} (\alpha)$</td>
<td>$\mu EPC_{i3} + AC_{i3} (\alpha)$</td>
<td>$\mu EPC_{i3} + AC_{i3} (\alpha)$</td>
</tr>
</tbody>
</table>

A well-defined platform reduces production costs by improving economies of scale and reducing the number of different components that are used. Suppose that a product family consists of unique modules, common modules, and EP modules as illustrated in Figure 6.2. The platform level is defined as the number of modules in the platform and consists of the common modules and the EP modules. An appropriate platform level for a product family is determined by minimizing the EP costs associated
with the EP modules. High levels of the platform (i.e., high commonality of the EP modules) decrease assembly and component costs while increasing EP loss costs. On the contrary, low platform levels (i.e., low commonality of EP modules) decrease EP loss costs while increasing assembly and component costs. As such, the appropriate platform level for the product family can be represented as a mathematical programming model in which EP costs are minimized. The mathematical model is discussed in the next section.

6.1.2 Mathematical Model for Platform Level Selection based on EP Cost

In this research, we consider a platform level problem as a component selection problem that is to determine EP values for unique or common components by minimizing the EP cost in a product family. The following assumptions are used to develop the model:

1. A platform can consist of common components and several EP components.
2. The objective is to determine the EP values of the EP components in the platform.
3. Each EP component can have several distinct EP values that can be purchased from various suppliers in the market.

4. Component costs for distinct EP values can be identified by the EP costs.

5. Target engineering requirements can be formulated as a function of a set of EP values.

6. When two or more EP components become a platform, assembly costs between them are zero.


The proposed mathematical model is focused on determining appropriate EP values in for EP components to design a platform. The component selection problem can be described as an integer programming problem using binary variables that are represented by decision variables for selecting components as a platform [73]. To formulate the mathematical model for platform level selection, the following notation is introduced:

\[
\begin{align*}
K & \quad \text{product index, } k = 1, 2, 3, \ldots, K \\
i, m & \quad \text{EP component index, } i = 1, 2, 3, \ldots, I, m = 1, 2, 3, \ldots, M \\
J_i, N_m & \quad \text{total EP number of EP component } i \text{ and } m \\
EP_{ij} & \quad j^{th} \text{ EP value of EP component } i \\
EPC_{kij} & \quad \text{component cost of } EP_{ij} \text{ in product } k \\
EPC^*_{ij} & \quad \text{platform cost of } EP_{ij} \\
AC_{ijn} & \quad \text{assembly cost between EP component } i j \text{ and } m n \\
AR_i^m & \quad \text{assembly relationship, } AR_i^m = 1 \text{ if EP component } i \text{ and } m \text{ are associated together, } AR_i^m = 0, \text{ otherwise} \\
TER_k & \quad \text{target engineering requirement of product } k \\
f_i(\{a_i\} \cup \{b_i\}, \forall i) & \quad \text{engineering requirement function consisting of a set of } EP_{ij} \text{ for product } k
\end{align*}
\]
Decision variables:

\[
y_{ij} = \begin{cases} 
1, & \text{if } j^{th} \text{ EP value of EP component } i \text{ in product } k \text{ is selected} \\
0, & \text{otherwise}
\end{cases}
\]

\[
y_{ij}^* = \begin{cases} 
1, & \text{if } j^{th} \text{ EP value of EP component } i \text{ is selected for the platform} \\
0, & \text{otherwise}
\end{cases}
\]

\[
y_{kmn} = \begin{cases} 
1, & \text{if EP } n \text{ is selected for EP component } m \text{ in product } k \text{ in the assembly} \\
0, & \text{otherwise}
\end{cases}
\]

\[
y_{kmn}^* = \begin{cases} 
1, & \text{if EP } n \text{ is selected for EP component } m \text{ as the platform in the assembly} \\
0, & \text{otherwise}
\end{cases}
\]

The mathematical model for platform level selection based on the EP cost model can be stated as follows:

Minimize \( Z = \sum_{k \in K} \sum_{i \in I} \sum_{j \in J} \sum_{k \in K} \sum_{j \in J} EPC_{kij} y_{ij} + \sum_{i \in I} \sum_{j \in J} EPC_{ij}^* y_{ij}^* \times K + \sum_{k \in K} \sum_{m \in M} \sum_{n \in N} \sum_{i \in I} \sum_{j \in J} (AC_{ij}^m AR_{ij}^m y_{kmn} + AC_{ij}^m AR_{ij}^m y_{kmn}^*) \) \hspace{1cm} (6.2)

subject to

\[
\sum_{k \in K} \sum_{j \in J} y_{ij} + K \sum_{j \in J} y_{ij}^* = K \text{ for all } i \hspace{1cm} (6.3)
\]

\[
\sum_{k \in K} \sum_{n \in N} y_{kmn} + K \sum_{n \in N} y_{kmn}^* = K \text{ for all } m \hspace{1cm} (6.4)
\]

\[ \text{TER}^k - f_i(\{a_j | a_j = \sum_{j \in J} EP_{ij} y_{ij}\} \cup \{b_j | b_j = \sum_{j \in J} EP_{ij} y_{ij}^*\}, \forall i) \leq 0 \text{ for all } k \hspace{1cm} (6.5) \]

\[
\sum_{k \in K} y_{kij} = 1, \sum_{k \in K} y_{kmn} = 1 \hspace{1cm} (6.6)
\]

\[
y_{kij}, y_{kmn}, y_{ij}^*, y_{kmn}^* = \{0, 1\} \hspace{1cm} (6.7)
\]
The objective function expressed in Eq. (6.2) is used to minimize the EP costs for determining an optimal platform level. The first term of Eq. (6.2) expresses unique EP component cost, and the second term represents common EP component costs. These two terms also include the EP loss costs of selecting higher EP values if a common EP does not meet the target engineering requirements. The third and fourth terms express assembly costs, including the cost reduction when EP components become a platform. Equations (6.3) and (6.4) indicate that EP components can be either unique or common, and Eq. (6.5) is a constraint ensuring that the engineering requirements of selected EP components are more than the targeted engineering requirements for all products. Equation (6.6) mandates that all corresponding EP components are unique unless they are common. To ensure that the decision variables are binary restrictions, Eq. (6.7) is defined.

If a product family is designed in a dynamic market environment where its information is decentralized, the complexity to solve the mathematical model is increased by the amount of distributed design information. Centralized control to solve the model causes high computation and communication costs. Decentralized control is an inevitable factor of managing product family design. In this research, therefore, we develop a dynamic multi-agent system based on negotiation mechanisms to solve this problem effectively. The next section introduces the dynamic multi-agent system in detail.
6.2 Dynamic Multi-Agent System

Software agents provide an ideal mechanism to realize information integration in design and manufacturing. Agents have been widely used in product design and can be used in product (and service) family design if developed properly. An agent has access to at least one and potentially many information sources and is able to collate and manipulate information obtained from these sources in order to answer queries posed by users and other information agents [94].

Most of the previous research efforts related to product design and multi-agent systems have been focused on the agents’ roles and tasks in a deterministic manufacturing environment. In product family design, a method to produce a variety of products should be considered for dynamic and various market segments. A multi-agent system can help achieve higher levels of flexibility, scalability, and adaptability in dynamic and distributed environments [41]. A multi-agent approach can be applied to develop an appropriate method to model a market-based product design system for the following reasons:

- A multi-agent system and market-based negotiation mechanisms are inherently distributed.
- Products can be designed with modules whose information is distributed in a market.
- Each module can be modeled as a self-interested agent – an autonomous decision-maker and a specific information holder at the same time.
A simple market-based negotiation mechanism can reduce the amount of communication in a distributed negotiation environment so that it increases the overall effectiveness in a product design system.

6.2.1 Electronic Market

In this work, an electronic market (e-market) is considered as a platform design environment for a product family that depends on how to determine platform design strategies representing alternative design methods in the market. A dynamic environment follows rudimentary e-market features such as business behaviors between buyers and sellers, dynamic pricing, and alternative selections [63, 91]. This e-market creates an environment where agents are economically motivated. The nature of an e-market allows economic agents (e.g., buyers and sellers) to freely enter or leave the e-market and negotiate with each other to obtain economic benefit. As shown in Figure 6.3, there are two types of agents in this e-market: (1) buyers and (2) sellers. Buyers are defined as auctioneers and sellers as bidders, and their goal is to maximize their own benefit. Depending on their strategy and market conditions, buyers and sellers purchase and provide products or services, respectively. Buyers can access all relevant sellers by querying information from them.
6.2.2 System Architecture

Agent-based decision-making based on agents’ roles and tasks can provide appropriate methods to solve product design problems [11, 67]. To facilitate the process of developing a market-based negotiation mechanism in this research, a dynamic multi-agent system (DMAS) as shown in Figure 6.4 is introduced based on five types of agents (refer to Figure 3.5): (1) a customer agent (CTA), (2) a coordinator agent (CDA), (3) a platform agent (PFA), (4) module agents (MDAs), and (5) component agents (CPAs). Knowledge related to product design development is stored in a knowledge base and used to define agents’ activities and tasks. The roles and knowledge of each agent were summarized in Table 3.1. And, knowledge for components and products is represented using the Techspecs Concept Ontology (TCO) as described in Section 4.2. MDAs can be configured by Techspecs. For example, Techspecs for EP components include: (a) capabilities: EP values, and (b) attributes: specifications, constraints, and quantities of EP components.
The main task in the DMAS is to design a product family by selecting appropriate EP values for a platform. Subtasks are used to determine EP values for supporting the main task. The CDA decomposes the main task into subtasks based on the product’s functions and assigns module design (subtasks) to MDAs by matching MDAs’ roles and tasks. MDAs fulfill the tasks requested from CDA and return the result to it. The number of MDAs is not fixed in the initial phase but is dynamically assigned based on the number of subtasks generated where each subtask designs a module to satisfy its one (or more) functional requirements.

When MDAs are generated by the CDA in the initial phase, they do not have information on the Techspecs of modules to fulfill their tasks. MDAs are initialized by determining necessary Techspecs based on the results of the first bidding with CPAs. During initialization, the CDA provides MDAs with knowledge for fulfilling their tasks.
Therefore, after first bidding (contact) with CPAs, MDAs can obtain their necessary Techspecs for designing modules. MDAs involved in designing an EP module provide alternative modules in terms of its EP values. After MDAs fulfill their tasks (the completion of a subtask), the information of the modules is translated into new knowledge for sharing and reuse. The next section discusses task decomposition for designing the modules.

### 6.2.3 Task Decomposition

A product is defined by its functions, and functions are achieved by a combination of modules. Suppose that a product family consists of \( l \) products, \( \mathbf{P}_F = (P^1, P^2, \ldots, P^l) \), and a product consists of \( n \) functions, \( P^i = (f_1, f_2, \ldots, f_n) \), \( f_i \in F \), where \( f_i \) is a function, and \( F \) is a set of functions in the functional level represented by TCO. A set of functions are used to assign subtasks for designing modules. Consequently, a main task for product family design is decomposed into subtasks: \( P^i = (T_1, T_2, \ldots, T_m) \), \( T_i \in T \) where \( T \) is a set of subtasks that consist of \( m \) tasks.

In the proposed DMAS, a main task can be decomposed into subtasks for designing modules to satisfy their functional requirements. For example, the task decomposition process in a product design system includes decomposing a product design (a high-level design task) into a number of assembly module designs (subtasks) which may be further decomposed into a number of component designs (sub-subtasks) [75]. For task decomposition, suppose new products are defined by the customer
requirements. Therefore, the number of products can be determined by the strategy of company and customers’ survey. The CDA identifies appropriate functions. Products to be developed are laid out based on the functions. The CDA decomposes the products into modules based on functions defined by TCO and assigns modules to MDAs so that it generates the same number of MDAs as the number of modules. If some modules are based on the common functions with different EP values, the CDA assigns these modules to one MDA. A MDA can configure various modules according to EP values. The information of EP values is defined in Techspecs that are determined by the first bidding with CPAs. Figure 6.5 shows the roles of CTA and CDA for task decomposition. In the next, an agent negotiation mechanism is discussed for determining EP values.

---

**Figure 6.5: The Roles of CTA and CDA for Task Decomposition**
6.3 Negotiation based on Market Mechanisms

As shown in Figure 6.6, there are three types of agents used in an e-market: (1) a platform agent (PFA), (2) module agents (MDAs), and (3) component agents (CPAs). The PFA and MDAs are defined as auctioneers (buyers) and the MDAs and CPAs as bidders (sellers), and their goal is to maximize their own individual benefit. MDAs can access all relevant component agents by querying information. We assume that the communication cost is small and that these agents only use cost information for negotiation to determine modules.

A module selection problem can be considered as a task allocation problem in which the right to design a module is given to a MDA. Auctions are useful techniques for allocating tasks to agents [46, 75, 94]. An auction-based negotiation can provide a set of negotiations to consider various design constraints that should be reflected in product
design. In the DMAS, a negotiation mechanism is employed to solve the task allocation problem between the PFA and MDAs.

MDAs configure their deals to negotiate with the PFA. If the subtask of a MDA is involved in designing an EP module, then based on the results of the first bidding with CPAs: \( \text{MDA}_i = \{D_i \mid D_j\} \), where \( i \) is MDA index (\( i^{th} \) MDA) and \( j \) is EP index (\( j^{th} \) EP values). A deal \( \delta_k \) is constructed by combining the deal sets of all MDAs for designing a platform. The PFA determines a feasible set of deals, \( \delta = \{\delta_1, \ldots, \delta_n\} \), for platform negotiation based on knowledge associated with designing a platform. The PFA uses a utility function of a MDA to determine the optimum EP values for a platform. Let \( T \) be a set of subtasks, \( EP \) a set of EP values, \( L \) a set of products, \( Q \) a set of module quantities, and \( S \) a set of all module agents in the system. \( T, EP, P, Q, \) and \( S \) are finite sets.

The utility \( u_i \) of a deal for MDA\(_i\) can be calculated by a utility function that is the difference between the cost of MDA\(_i\) doing subtask \( T_i \) that is initially assigned for designing unique modules, \( c(T_i) \), and the cost of MDA\(_i\) doing a deal \( \delta_k \), \( e_i(\delta_k) \):

\[
u_i(\delta) = c(T_i) - e_i(\delta_k) \tag{6.8}\]

where \( c(T_i) \) is estimated by an expected cost function for unique modules:

\[
c^i : T \times EP \times S \mapsto \mathbb{R}. \]

The real number of \( c^i(t, ep, s) \) represents the cost of module agent \( s \) paying EP cost \( (ep) \) for performing subtask \( t \). For example, if components are designed as a unique module, then the expected cost can be determined by component cost and assembly cost and given as:

\[
\eta \sum_{i \in l} ep_i = \eta(\sum_{i \in l} EPC_i + AC_i) \tag{6.9}\]
where $l$ is the number of products, and $\eta$ is a factor for overhead. On the other hand, $e_i(\delta_k)$ is estimated by an expected deal cost function: $e^i_{\delta_k}: T \times EP \times L \times Q \times S \rightarrow \mathbb{R}$.

Hence, the real number of $e^i_{\delta_k}(t, ep, p, q, s)$ represents the cost of module agent $s$ paying $ep$ at quantity $q$ for performing a deal $(\delta_k)$ of subtask $t$ for a platform for $l$ products. For example, if components are designed as common modules, then the expected deal cost for $\delta_k$, based on EP costs, is:

$$e^i_{\delta_k}(t, ep, l, q, s) = \eta \times ep(l \times q) \bigg|_{\delta_k} = \eta \left( \frac{EPC(\delta_k)}{l \times q} + AC(\delta_k) + D(\delta_k) \right)$$

(6.10)

where $p$ is a subtask weighting function as follows:

$$l = \begin{cases} 1, & \text{if module is unique} \\ L, & \text{otherwise} \end{cases}$$

(6.11)

where $l$ is the number of products, and $q$ is a quantity function as follows:

$$q = \begin{cases} 1, & \text{if module is unique} \\ \chi, & \text{otherwise} \end{cases}$$

(6.12)

where $\chi$ is a volume discount factor. For a given set of products, $c(T_i)$ has a constant value, but the value of $e_i(\delta_k)$ varies depending on choosing a deal for a platform.

The PFA uses this utility function, Eq. (6.8), to choose proper EP values for the platform. If a subtask is to design unique modules, its utility function has zero value since the expected deal cost is the same as the cost of doing the subtask for unique modules. On the contrary, if a subtask is to design a common module, its utility function has positive values since the expected deal cost is less than the cost of doing the subtask for all unique modules. For example, suppose that subtask $T_i$ is to design a common module $(\delta_k)$. Its
expected deal cost, \( e_i(\delta_k) \), decreases since both \( p \) and \( q \) have more than 1, which results in high utility values.

### 6.3.1 Negotiation Protocol between PFA and MDAs

Agents in negotiation use protocols to define the rules of interaction. The protocol for platform design is developed based on an auction mechanism that guarantees convergence to a solution [89]. For one PFA and two MDAs, the rules of the protocol are developed as follows. After allocating tasks and bidding with CPAs, the MDAs propose deals based on their Techspecs. Then, the PFA configures a set of deals for negotiation and sends the MDAs the set of deals to calculate their utility values. The process of negotiation is illustrated in Figure 6.7.

1. Allocate task, \( T_1, T_2 \)
2. Define \( c(T) \), Deal \( (D_{11}, D_{21}, D_{22}) \) based on Techspecs, and Send
3. Configure a set of deals
4. Send a set of deals to MDA
5. Calculate utilities for all deals
6. Send utilities to PFA
7. Make a decision

Figure 6.7: Negotiation Process for Two MDAs: An Example
If there is a conflict between any of the MDAs, then risk value analysis is used to reach an agreement. For example, suppose that $\delta_1 = (D_{11}, D_{21})$ and $\delta_2 = (D_{11}, D_{22})$ are a set of deals for a platform. If different deals are selected to maximize the utility value for different subtasks, such as $T_1 : u_1(\delta_1) > u_1(\delta_2)$ and $T_2 : u_2(\delta_1) < u_2(\delta_2)$, then we need to analyze the risk value of performing those subtasks for a platform. In this work, $\text{risk}_i^{\delta}$ is defined as the average loss of the utility value when a deal $\delta$ is not selected and an alternative deal is selected, and is represented as:

$$\text{risk}_i^{\delta} = \frac{\text{avg} \{u_i(\delta_l) - u_i(\delta_k)\}}{\forall \delta}, \ k \neq l,$$

(6.13)

To minimize risk, the PFA selects the minimum average risk value for each deal. If the same average risk values are encountered, then the deal with the higher utility value is chosen. As an example, given the values of subtasks’ utility and the risk values in Table 6.2, Deal 2, $\delta_2$, would be selected.

<table>
<thead>
<tr>
<th>Task</th>
<th>Deal</th>
<th>$\delta_1$</th>
<th>$\delta_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1$</td>
<td>3 (-1)</td>
<td>2 (1)</td>
<td></td>
</tr>
<tr>
<td>$T_2$</td>
<td>1 (2)</td>
<td>3 (-2)</td>
<td></td>
</tr>
<tr>
<td>Average risk</td>
<td>0.5</td>
<td>-0.5</td>
<td></td>
</tr>
</tbody>
</table>

### 6.3.2 Negotiation Mechanism between MDAs and CPAs

In a market when components in a module are offered through individual auctions, a buyer selects a bundle of components to minimize the cost of a combination
of components. Combinatorial auctions provide an optimal solution by permitting bids on bundles of components [95]. In the proposed DMAS, negotiation between MDAs and CPAs uses a trade protocol based on combinatorial auction mechanisms that use a bidding language for facilitating a set of items, where CPAs can submit bids for combinations of components.

Suppose that one unit of each component is offered in an auction. Let \( m \) be the number of unique components for a deal and \( t \) be the total number of CPAs (bidders). If we can consider that a unique CPA provides one bundle (module or combination of components), then the number of CPAs is the same as the number of bids, \( t \). A mathematical model represents this as follows [95]:

\[
\text{Minimize } Z = \sum_{j=1}^{t} p_j x_j \tag{6.14}
\]

Subject to

\[
\sum_{j=1}^{t} w_j x_j \leq 1 \tag{6.15}
\]

\[
x_j = 0, 1 \quad j = 1, \ldots, t \tag{6.16}
\]

where \( x_j \) is a binary variable, representing whether bundle \( j \) gets traded; \( p_j \) is the bid price for bundle \( j \); and \( w_j \) is a vector of size \( m \) where \((w_j)_k \) is one if component \( k \) is part of bundle \( w_j \) and zero otherwise. The result of this model provides a MDA with an optimal solution to fulfill its tasks.

For example, suppose that a module consists of three components: A, B, and C \((m=3)\). Consider the following bids \( x_1, \ldots, x_5 \): bid = ([Components], price), where the
number of bidders is 5. The bids are: \( x_1 = ([A], 15) \), \( x_2 = ([B], 15) \), \( x_3 = ([C], 15) \), \( x_4 = ([A, C], 25) \), \( x_5 = ([A, B, C], 45) \). The vectors of bundles are: \( w_1 = (1, 0, 0) \), \( w_2 = (0, 1, 0) \), \( w_3 = (0, 0, 1) \), \( w_4 = (1, 0, 1) \), \( w_5 = (1, 1, 1) \). The resulting constraints are:

\[
\begin{bmatrix}
1 \\
0 \\
0 \\
0 \\
0 \\
\end{bmatrix} x_1 + \begin{bmatrix}
0 \\
1 \\
0 \\
0 \\
1 \\
\end{bmatrix} x_2 + \begin{bmatrix}
0 \\
0 \\
1 \\
1 \\
1 \\
\end{bmatrix} x_3 + \begin{bmatrix}
1 \\
0 \\
0 \\
1 \\
1 \\
\end{bmatrix} x_4 + \begin{bmatrix}
1 \\
1 \\
1 \\
1 \\
1 \\
\end{bmatrix} x_5 \leq 1
\]  \quad (6.17)

The optimal solution is \( x_2 = x_4 = 1 \), \( x_1 = x_3 = x_5 = 0 \), and \( Z = 40 \).

At this stage, agents negotiate with each other, using pre-defined rule-based strategies that consist of combinatorial auction protocols. A winner (CPA) is decided by the result of negotiation between a CDA and MDAs. If the MDA is selected as a module for a platform, then CPAs related to the MDA have the right of providing their components. First, MDAs announce the necessary components to perform their tasks to the CPAs. The contents of the bids from the CPAs will be the information from the MDAs’ Techspecs. Based on a combinatorial auction mechanism, each MDA generates an initial solution and sends it to the PFA. During negotiation between the PFA and the MDAs, the MDA can continuously negotiate with the CPA based on the changing results of the deal with the PFA.

### 6.3.3 Case Study: Power Tool Family

The objective in this case study is to determine an appropriate platform for the power tool family introduced in Section 4.2.6 using the negotiation mechanism subject to a certain market environment. The case study focuses on the negotiation between the
PFA and MDAs to develop a new platform. Currently, these products have common modules related to electrical components as the platform. Using the proposed DMAS, we develop a new platform that consists of the common and the EP modules related to customer needs (CNs) and the market situation. We assume that the CNs of tools are as defined in Table 6.4, and functional requirements (FRs) are generated from CNs and the Function-Component Mapping Matrix (FCM) in the Design Repository. CTA performs the analysis of CNs to understand customer intention and map the CNs to FRs. Table 6.3 shows the results of analyzing the CNs and FRs for the power tool family.

---

10 http://function.basiceng.umr.edu/repository
<table>
<thead>
<tr>
<th>Tool</th>
<th>Customer needs</th>
<th>Functional requirements</th>
<th>Engineering parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circular saw</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Circle cutting</td>
<td>- Convert e.e. to m.e.</td>
<td>- Current</td>
<td></td>
</tr>
<tr>
<td>- Electric method</td>
<td>- Import e.e., h.e.</td>
<td>- Torque</td>
<td></td>
</tr>
<tr>
<td>- Lightweight and portable</td>
<td>- Change speed</td>
<td>- Gear ratio</td>
<td></td>
</tr>
<tr>
<td>- Safety</td>
<td>- Actuate e.e.</td>
<td>- RPM</td>
<td></td>
</tr>
<tr>
<td>- Varying speed</td>
<td>- Store e.e., m.e., h.e., solid</td>
<td>- Weight</td>
<td></td>
</tr>
<tr>
<td>- Variable cut angle</td>
<td>- Transfer e.e., m.e., force</td>
<td>- Size</td>
<td></td>
</tr>
<tr>
<td>Jig saw</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Cutting</td>
<td>- Convert e.e. to m.e.1</td>
<td>- Current</td>
<td></td>
</tr>
<tr>
<td>- Electric method</td>
<td>- Convert m.e.1 to m.e.2</td>
<td>- Torque</td>
<td></td>
</tr>
<tr>
<td>- Lightweight and portable</td>
<td>- Import e.e., h.e., solid, hand</td>
<td>- Gear ratio</td>
<td></td>
</tr>
<tr>
<td>- Safety</td>
<td>- Change speed</td>
<td>- RPM</td>
<td></td>
</tr>
<tr>
<td>- Varying speed</td>
<td>- Actuate e.e.</td>
<td>- Weight</td>
<td></td>
</tr>
<tr>
<td>- Quick blade change mechanism</td>
<td>- Store e.e., m.e., h.e., solid</td>
<td>- Size</td>
<td></td>
</tr>
<tr>
<td>Drill</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Drilling</td>
<td>- Convert e.e. to m.e.1</td>
<td>- Current</td>
<td></td>
</tr>
<tr>
<td>- Electric method</td>
<td>- Import e.e., h.e., solid, hand</td>
<td>- Torque</td>
<td></td>
</tr>
<tr>
<td>- Lightweight and portable</td>
<td>- Change speed</td>
<td>- Gear ratio</td>
<td></td>
</tr>
<tr>
<td>- Safety</td>
<td>- Actuate e.e.</td>
<td>- RPM</td>
<td></td>
</tr>
<tr>
<td>- Varying speed</td>
<td>- Store e.e., m.e., h.e., solid</td>
<td>- Weight</td>
<td></td>
</tr>
<tr>
<td>- Quick drill bite</td>
<td>- Transfer e.e., m.e., force</td>
<td>- Size</td>
<td></td>
</tr>
<tr>
<td>- Change mechanism</td>
<td>- Separate solid</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Variable bite size</td>
<td>- Secure solid</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- High speed motor</td>
<td>- Export solid</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sander</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Smooth surface</td>
<td>- Convert e.e. to m.e.1</td>
<td>- Current</td>
<td></td>
</tr>
<tr>
<td>- Electric method</td>
<td>- Convert m.e.1 to m.e.2</td>
<td>- Torque</td>
<td></td>
</tr>
<tr>
<td>- Lightweight and portable</td>
<td>- Import e.e., h.e., solid, hand</td>
<td>- Gear ratio</td>
<td></td>
</tr>
<tr>
<td>- Safety</td>
<td>- Change speed</td>
<td>- RPM</td>
<td></td>
</tr>
<tr>
<td>- Ease in switching ON/OFF</td>
<td>- Actuate e.e.</td>
<td>- Vibration strength</td>
<td></td>
</tr>
<tr>
<td>- Should not heat up fast</td>
<td>- Store e.e., m.e., h.e., solid</td>
<td>- Weight</td>
<td></td>
</tr>
<tr>
<td>- Variable speed</td>
<td>- Transfer e.e., m.e., force</td>
<td>- Size</td>
<td></td>
</tr>
<tr>
<td>- Safety</td>
<td>- Separate solid</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Ease in switching ON/OFF</td>
<td>- Secure solid</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Variable size nails</td>
<td>- Export solid</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nailer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Nailer</td>
<td>- Convert e.e. to m.e.1</td>
<td>- Current</td>
<td></td>
</tr>
<tr>
<td>- Electric method</td>
<td>- Convert m.e.1 to m.e.2</td>
<td>- Torque</td>
<td></td>
</tr>
<tr>
<td>- Lightweight and portable</td>
<td>- Import e.e., h.e., solid, hand</td>
<td>- Gear ratio</td>
<td></td>
</tr>
<tr>
<td>- Easy loading</td>
<td>- Change speed</td>
<td>- RPM</td>
<td></td>
</tr>
<tr>
<td>- Safety</td>
<td>- Actuate e.e.</td>
<td>- Hit strength</td>
<td></td>
</tr>
<tr>
<td>- Variable size nails</td>
<td>- Store e.e., m.e., h.e., solid</td>
<td>- Weight</td>
<td></td>
</tr>
<tr>
<td>- High capacity (less load)</td>
<td>- Transfer e.e., m.e., force</td>
<td>- Size</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Separate solid</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Secure solid</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Export solid</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1.e.e.: electronic energy, 2.m.e.: mechanical energy, 3.h.e.: human energy
A. Task Decomposition for MDAs

In the proposed DMAS, FRs are subtasks to design modules as stated in Section 6.2.2. The subtasks for MDAs are identified as modules based on their functional structure models (refer to Appendix A). Table 6.4 shows modules for the tool family including MDAs and their target engineering parameters in EP modules. There are 16 subtasks for developing the new product family: one common module, two EP modules, and 13 unique modules. In the case study, we assume that the engineering parameters of the products are defined as the torque of a motor and the current of a battery. The CDA assigns these modules to MDAs as shown in Table 6.4. MDAs are initialized based on the Techspecs of the CDA as shown in Table 6.5. For example, the subtask of MDA 1 is to design an electronic module that consists of three components: a switch, a battery terminal, and wires. The Techspecs of MDA 1 include size, weight, type, cost, and quantity.

Table 6.4: The Results of Tasks Decomposition

<table>
<thead>
<tr>
<th>Product</th>
<th>Common module</th>
<th>EP module</th>
<th>Unique module (MDA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circular saw</td>
<td>Electronic (MDA1)</td>
<td>Motor (MDA2, 12 N.m.), Battery (MDA3, 5.0 A)</td>
<td>Blade mounting (MDA4), Input1(MDA5)</td>
</tr>
<tr>
<td>Jig saw</td>
<td>Electronic (MDA1)</td>
<td>Motor (MDA2, 12 N.m.), Battery (MDA3, 5.0 A)</td>
<td>Conversion1 (MDA6), Blade mounting (MDA7), Input2 (MDA8)</td>
</tr>
<tr>
<td>Sander</td>
<td>Electronic (MDA1)</td>
<td>Motor (MDA2, 7 N.m.), Battery (MDA3, 5.0 A)</td>
<td>Conversion2 (MDA9), Sander mounting (MDA10), Input3 (MDA11)</td>
</tr>
<tr>
<td>Drill</td>
<td>Electronic (MDA1)</td>
<td>Motor (MDA2, 15 N.m.), Battery (MDA3, 7.0 A)</td>
<td>Drill mounting (MDA12)</td>
</tr>
<tr>
<td>Brad nailer</td>
<td>Electronic (MDA1)</td>
<td>Motor (MDA2, 15 N.m.), Battery (MDA3, 7.0 A)</td>
<td>Nail Hitter (MDA14), Magazine assembly (MDA15), Input5 (MDA16)</td>
</tr>
</tbody>
</table>
In this section, we demonstrate how to use the market-based negotiation mechanism to determine a new platform for the power tool family. The negotiation process proceeds as follows.

### B. Negotiations between PFA and MDAs

In this section, we demonstrate how to use the market-based negotiation mechanism to determine a new platform for the power tool family. The negotiation process proceeds as follows.

#### Step 1) Task allocation

Based on the initialization of Techspecs as in Table 6.6, MDAs can obtain information on Techspecs to fulfill their tasks from the CPAs by an auction. We generate the numerical data based on unit cost that is dependent on components’ EP values. Suppose that Table 6.7 illustrates the result of the first bidding with the CPAs in a market environment. The negotiation between the CDA and MDAs for the platform of this family is conducted once the first bidding is initiated.
Step 2) Deal configuration for MDAs

Common and unique module design can be determined from the first bidding based on market mechanisms to minimize total production cost. Since MDA2 and MDA3 have EP modules, we suppose that they are involved in designing the platform. Therefore, appropriate EP values for EP modules can be determined by negotiation to design the platform. MDA2 and MDA3 can configure their deals to negotiate with the PFA using their Techspecs. These agents generate deals to meet the target engineering parameters in the knowledgebase. For example, from Table 6.7, a deal for MDA2 consists of three components: Motor A, Gear A, and Shaft A. The torque for the deal is 7.5 N.m. (torque 3 N.m \times gear ratio 2.5), which satisfies the target engineering parameter for the sander. The deals of MDA2 and MDA3 are shown in Table 6.7

<table>
<thead>
<tr>
<th>MDA</th>
<th>Component</th>
<th>Eng. parameter</th>
<th>Weight (g)</th>
<th>Size</th>
<th>Cost</th>
<th>Ass’y Cost</th>
<th>CPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDA2</td>
<td>A Motor</td>
<td>Torque 3 N.m.</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>CAP1</td>
</tr>
<tr>
<td></td>
<td>B Motor</td>
<td>Torque 5 N.m.</td>
<td>6</td>
<td>3</td>
<td>7</td>
<td>2</td>
<td>CAP2</td>
</tr>
<tr>
<td></td>
<td>A Gear</td>
<td>Gear ratio 2.5</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>CAP3</td>
</tr>
<tr>
<td></td>
<td>B Gear</td>
<td>Gear ratio 4.0</td>
<td>4</td>
<td>2</td>
<td>6</td>
<td>1</td>
<td>CAP4</td>
</tr>
<tr>
<td></td>
<td>Shaft</td>
<td>-</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>CAP5</td>
</tr>
<tr>
<td>MDA3</td>
<td>A Battery</td>
<td>Current 5.0 A</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>CAP6</td>
</tr>
<tr>
<td></td>
<td>B Battery</td>
<td>Current 7.0 A</td>
<td>4</td>
<td>2</td>
<td>6</td>
<td>1</td>
<td>CAP7</td>
</tr>
</tbody>
</table>

Table 6.7: The Result of the First Bidding with CPAs for EP values

<table>
<thead>
<tr>
<th>Agent</th>
<th>Deal (T)</th>
<th>Component</th>
<th>EP level</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDA2</td>
<td>D_{21}</td>
<td>A,A,A</td>
<td>7.5 N.m.</td>
</tr>
<tr>
<td></td>
<td>D_{22}</td>
<td>A,B,A</td>
<td>12 N.m.</td>
</tr>
<tr>
<td></td>
<td>D_{23}</td>
<td>B,A,A</td>
<td>12.5 N.m.</td>
</tr>
<tr>
<td></td>
<td>D_{24}</td>
<td>B,B,A</td>
<td>20 N.m.</td>
</tr>
<tr>
<td>MDA3</td>
<td>D_{31}</td>
<td>A</td>
<td>5.0 A</td>
</tr>
<tr>
<td></td>
<td>D_{32}</td>
<td>B</td>
<td>7.0 A</td>
</tr>
</tbody>
</table>

Table 6.7: Deals for a Negotiation with PFA
Step 3) Deal set identification for a platform

The PFA identifies a set of deals for the platform by a combination of the deals of the MDAs, which is shown in Table 6.8. To select EP values for the platform, the PFA uses the utility value of the set of deals. Assume that the EP costs, subtask weighting function, and quantity function are as given in Table 6.8. Then the EP costs can be calculated by the EP cost model given by Eq. (6.1). We assume that the loss cost of EP is defined as additional component cost, which is 1 unit and 2 units for torque and current, respectively.

<table>
<thead>
<tr>
<th>Deal (platform)</th>
<th>Deal (T)</th>
<th>$T_2$-EP cost</th>
<th>$T_3$-EP cost</th>
<th>$p$</th>
<th>$q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\delta_1$</td>
<td>$D_{21}, D_{31}$</td>
<td>22.2</td>
<td>7.4</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>$\delta_2$</td>
<td>$D_{21}, D_{32}$</td>
<td>22.2</td>
<td>7</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>$\delta_3$</td>
<td>$D_{22}, D_{31}$</td>
<td>20.2</td>
<td>7.4</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>$\delta_4$</td>
<td>$D_{22}, D_{32}$</td>
<td>20.2</td>
<td>7</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>$\delta_5$</td>
<td>$D_{23}, D_{32}$</td>
<td>20</td>
<td>7</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>$\delta_6$</td>
<td>$D_{24}, D_{32}$</td>
<td>21</td>
<td>7</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

Step 4) Deal cost calculation

After identifying deal sets, the PFA calculates the cost of subtasks based on the minimum cost of each deal. The subtasks also should satisfy the constraints when the deals are selected. After the PFA selects the deal for each product for a subtask, the cost of the subtask is obtained by averaging the cost of each deal. For example, $D_{21}$ provides minimum cost for satisfying the constraint of the sander and has a module cost of 13 (5+4+4) and an assembly cost of 4 (2+1+1). We assumed that the factor for overhead cost is 2. The cost of $D_{21}$ in Subtask 2 is represented by $c^2(2,17,2)$ and is 34, which was
calculated by Eq. (6.9). As shown in Table 6.9, the average cost of Subtask 2 is 38.8. We can obtain the cost of Subtask 3 using the same processes. To estimate the cost of a subtask, we assumed that the predicted quantities for the new products were the same.

<table>
<thead>
<tr>
<th>Product</th>
<th>Torque constraint</th>
<th>Module cost</th>
<th>Deal cost</th>
<th>(c(T_2))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circular saw</td>
<td>12 N.m.</td>
<td>19 (A, B, A)</td>
<td>38</td>
<td></td>
</tr>
<tr>
<td>Jig saw</td>
<td>12 N.m.</td>
<td>19 (A, B, A)</td>
<td>38</td>
<td></td>
</tr>
<tr>
<td>Sander</td>
<td>7 N.m.</td>
<td>17 (A, A, A)</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>Drill</td>
<td>15 N.m.</td>
<td>21 (B, B, A)</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>Brad nailer</td>
<td>15 N.m.</td>
<td>21 (B, B, A)</td>
<td>42</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.9: The Constraints and Cost of Subtask 2 for Products

Step 5) Utility calculation and decision making

Table 6.10 represents the result of the negotiation between the PFA and the MDAs. The utility and the values of the deal were calculated based on Eqs. (6.8) and (6.10), respectively. For example, the value of deal \(\delta_1\) in Subtask 2 is represented as \(e^2_{\delta_1}(2,22.2,2,3,2)\) and is 3.7. The utility of deal \(\delta_1\) in Subtask 2 is 35.1 (= 38.8 – 3.7).

Based on the result of Table 6.10 and the negotiation protocol, the CDA selects Deal 5 as the platform. In this case, since the selected deal set, \(\delta_5\), consists of the same deals, \(D_{23}\) and \(D_{32}\) with respect to the Subtasks 2 and 3, the risk value for the deals is not necessary for decision-making. Deal 5 can provide the minimum cost when developing a new platform. Therefore, the new platform can be designed as an electronic module, a motor module with torque 3 N.m. and gear ratio 2.5, and a battery module with current 7.0 A.
Through the case study, the proposed market-based negotiation mechanism was demonstrated to determine the EP-value for EP module to select appropriate modules for the platform. Therefore, the negotiation mechanism in DMAS can facilitate product family design in various dynamic e-market environments. In the next section, a game theoretic approach is proposed to help determine platform design strategies in an uncertain market environment.

### 6.4 Game Theoretic Approach for Strategic Platform Design

The objective in this section is to propose an approach for strategic platform design in a product family using concepts from game theory to model situations in uncertain e-market environments. Module-based platform design is identified by introducing unique modules, common modules, and engineering parameter (EP) modules for product family design. In this section, a module selection problem is considered as a strategic game with incomplete information that is described by products’ market share.

<table>
<thead>
<tr>
<th>Task</th>
<th>Deal (platform)</th>
<th>$c(T_i)$</th>
<th>$e(\delta_i)$</th>
<th>$\mu_i$</th>
<th>Deal (task)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>$\delta_1$</td>
<td>38.8</td>
<td>3.7</td>
<td>35.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\delta_2$</td>
<td></td>
<td>3.7</td>
<td>35.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\delta_3$</td>
<td></td>
<td>3.37</td>
<td>35.43</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\delta_4$</td>
<td></td>
<td>3.37</td>
<td>35.43</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\delta_5$</td>
<td>3.33</td>
<td><strong>35.47</strong></td>
<td></td>
<td>$D_{33}$</td>
</tr>
<tr>
<td></td>
<td>$\delta_6$</td>
<td></td>
<td>3.5</td>
<td>35.3</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>$\delta_1$</td>
<td>11.2</td>
<td>1.23</td>
<td>9.97</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\delta_2$</td>
<td></td>
<td>1.17</td>
<td>10.03</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\delta_3$</td>
<td></td>
<td>1.23</td>
<td>9.97</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\delta_4$</td>
<td></td>
<td>1.17</td>
<td>10.03</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\delta_5$</td>
<td>1.17</td>
<td><strong>10.03</strong></td>
<td></td>
<td>$D_{32}$</td>
</tr>
</tbody>
</table>

Table 6.10: The Result of Negotiation between CDA and STA2, 3
ratios and customer’s preferences. A Bayesian game is designed to model situations wherein some of players have incomplete information or uncertain characteristics for the other players [23, 62]. The players’ uncertainty about each other is described as a set of possible types (or type spaces). A Bayesian game is employed to model uncertainty situations regarding market environments and decide strategic equilibrium solutions for selecting modules in the product family being designed. The proposed approach can be applied to a method for agent-based decision-making between module agents and product market agents as shown in Figure 3.5 by defining their roles and knowledge.

6.4.1 A Bayesian Game with Incomplete Information

In this section, the normal-form representation of a static Bayesian game is reviewed [23]:

**Definition 6.1**: The normal-form representation of an $n$-players static Bayesian game consists of the players’
- Action spaces, $A_1, \ldots, A_n$
- Type spaces, $T_1, \ldots, T_n$
- Beliefs, $b_1, \ldots, b_n$
- Payoff functions, $u_1, \ldots, u_n$

where player $i$’s type, $t_i$, is privately known by player $i$, determines player $i$’s payoff function, $u_i(a_1, \ldots, a_n; t_i)$, and is the member of the possible types, $T_i$. Player $i$’s belief $b_i(t_{-i} | t_i)$ describes $i$’s uncertainty about the other $n-1$ players’ possible types, $t_{-i}$, given $i$’s own type, $t_i$. We denote this game by $G = \{A_1, \ldots, A_n; T_1, \ldots, T_n; b_1, \ldots, b_n; u_1, \ldots, u_n\}$. 

For an equilibrium concept for static Bayesian games, the players’ strategy spaces in such a game are defined by [23]:
Definition 6.2: In the static Bayesian game \( G = \{ A_1, \ldots, A_n; T_1, \ldots, T_n; b_1, \ldots, b_n; u_1, \ldots, u_n \} \), a strategy for player \( i \) is a function \( s_i(t_i) \), where for each type \( t_i \) in \( T_i \), \( s_i(t_i) \) specifies the action from the feasible set \( A_i \) that type \( t_i \) would choose if drawn by nature.

In a Bayesian game, each player’s strategy must be a best response to the other players’ strategies. Therefore, given the definition of a strategy in a Bayesian game, a Bayesian Nash equilibrium is defined formally as [23]:

Definition 6.3: In the static Bayesian game \( G = \{ A_1, \ldots, A_n; T_1, \ldots, T_n; b_1, \ldots, b_n; u_1, \ldots, u_n \} \), the strategies \( s^* = (s_1^*, \ldots, s_n^*) \) are a (pure-strategy) Bayesian Nash equilibrium if for each player \( i \) and for each of \( i \)'s types \( t_i \) in \( T_i \), \( s_i^*(t_i) \) solves

\[
\max_{a_i \in A_i} \sum_{t_{i-1} \in T_{i-1}} u_i(s_i^*(t_i), \ldots, s_{i-1}^*(t_{i-1}), a_i, s_{i+1}^*(t_{i+1}), \ldots, s_n^*(t_n); t_i) b_i(t_{i-1} | t_i).
\] (6.17)

That is, no player wants to change his or her strategy, even if the change involves only one action by one type.

In the next section, the expected strategy cost and the proposed Bayesian game for product family design are discussed in detail.

6.4.2 Cost of Expected Strategies

A platform designer determines a feasible set of strategies for the platform based on his/her design knowledge. The strategies are represented as alternative design methods and can be constructed by combining components in EP modules for a platform. Let \( S \) be a set of strategies, \( EP \) a set of EP values, \( L \) a set of products in a product family, and \( Q \) a set of module quantities. \( S, EP, L, \) and \( Q \) are finite sets.
The expected strategy cost, \( c(s_i) \), for designer’s strategy \( s_i \) \((i = 1, \ldots, S)\) is estimated by an expected strategy cost function: \( f^i : S \times EP \times L \times Q \mapsto \mathbb{R} \). Hence, the real number of \( f^i(s_i, ep, l, q) \) represents the cost of strategy paying \( ep \) at quantity \( q \) for performing strategy \( s_i \) for a platform from \( l \) products. For example, the expected strategy cost for \( s \) can be determined based on EP costs as:

\[
f^i(s_i, ep, l, q) = \sum \frac{(EPC_i + AC_i + D_i)}{l \times q}
\]

where \( l \) is a strategy weighting function as follows:

\[
l = \begin{cases} 
1, \text{ if module is unique} \\ 
L, \text{ otherwise}
\end{cases}
\]

and \( q \) is a quantity function as follows:

\[
q = \begin{cases} 
1, \text{ if module is unique} \\ 
\chi, \text{ otherwise}
\end{cases}
\]

where \( \chi \) is a volume discount factor or market share ratio. For a given set of products, the value of \( c(s_i) \) varies depending on the strategy for platform design. The next section discusses a Bayesian game model for determining a platform design strategy.

### 6.4.3 Bayesian Game Model for Strategic Product Platform Design

Consider the following module selection problem for platform design in a dynamic and uncertain market environment. There are two agents: (1) a designer who has module design strategies for a product family and (2) a customer who has prices for a module. The designer provides a module with the cost \( c \), and the customer pays the price
v for the module. The designer’s cost and the customer’s price are dependent on the market share ratio of the products and customer’s preference, respectively. The market share ratio and the customer’s preference are assumed to be independently and uniformly distributed based on their market information. The cost and the price are constrained to be non-negative. If the market share ratio is greater than or equal to the customer’s preference, then the module will be produced at a price equal to the average of the module’s cost and the customer’s price; otherwise, the module will be not produced. Finally, let us assume that the players are risk-neutral for their payoffs. Each player knows his or her own payoff function but may be uncertain about the other player’s payoff functions.

In order to formulate the proposed scenario as a Bayesian game, we must first identify the action spaces, the type spaces, the beliefs, and the payoff functions [23]. In this section, Agent 1 is a designer who knows module design strategies for a platform. Agent 1’s action is to select a strategy among design strategies that are all possible combinations for EP module design. The set of actions, $A_1 = \{a_{1,1}, a_{1,2}, \ldots, a_{1,n_1}\}$, for Agent 1 are represented by the design strategies that can be developed by a designer for products. The set of types, $T_1 = \{t_{1,1}, t_{1,2}, \ldots, t_{1,m_1}\}$, are the values of the market share ratio for the products including the module, and the values are obtained from a uniform distribution on $[0,1]$. Because the values of the market share ratios are independent, Agent 1 believes that the probability, $b_1$, is uniformly distributed on $[0,1]$. Meanwhile, Agent 2 is a customer who wants to buy a module with a market price. Agent 2’s action is to determine the price of the module. The set of actions for Agent 2, $A_2 = \{a_{2,1}, a_{2,2}, \ldots, a_{2,n_2}\}$ constitutes
modules’ prices based on the market price. In this section, we define the market price as 
\((2 \times w_i c(s_i))\), where \(w_i\) is the proportion of the number of products that satisfy EP
constraints by strategy \(i\) in a family \((0 < w_i \leq 1, \ i \in n_i)\). The set of types of Agent 2,
\(T_2 = \{t_{2,1}, t_{2,2}, \ldots, t_{2,m_2}\}\), are represented by customer’s preferences, and the values of the
preferences can be obtained from a uniform distribution on \([0,1]\). Because the values of
the customer’s preference are independent, Agent 2 believes that the probability, \(b_2\), is
uniformly distributed on \([0,1]\). \(A_1, A_2, T_1, \) and \(T_2\) are finite sets that are defined by the
number of \(n_1, n_2, m_1, \) and \(m_2,\) respectively. Therefore, Agent 1 may be uncertain about
the Agent 2’s payoff functions, since Agent 1 may be uncertain about the types of Agent
2, denoted by \(t_{-1}\). In this game, the probability distribution \(b_1(t_{-1} | t_1)\) is defined as Agent
1’s belief about Agent 2’s types, \(t_{-1}\), given Agent 1’s knowledge based on type, \(t_1\).
According to the proposed scenario, Agent 1 and Agent 2 can two possible payoff
functions based on their selected types. The two agents’ payoff functions are given by:
\[
\begin{align*}
  u_1(a_{1}^{*}, a_{2}^{*}; t_1) & = (c + v)/2 - c, \text{ if } t_1 \geq t_2 \\
  u_2(a_{1}^{*}, a_{2}^{*}; t_2) & = (c + v)/2 - v, \text{ if } t_1 \geq t_2 \\
  u_1(a_{1}^{*}, a_{2}^{*}; t_1) & = u_2(a_{1}^{*}, a_{2}^{*}; t_2) = 0, \text{ if } t_1 < t_2
\end{align*}
\]  
(6.21)  
(6.22)  
(6.23)

where \(c\) is the expected cost based on \(a_1^{*}\) and is calculated by the expected strategy cost
mentioned in Section 6.4.2, and \(v\) is the price of the module based on \(a_2^{*}\) and is calculated
by (the market price × \(t_2\)). Formally, this game is denoted by
\(G = \{A_1, A_2; T_1, T_2; b_1, b_2; u_1, u_2\}\) as shown in Figure 6.8.
In this scenario, Agent 1 will try to seek a module that provides more profits as a platform in uncertain customers’ preferences based on minimizing the expected strategy cost in various market share ratios. Otherwise, Agent 2 wants to buy a module that provides more functions in a product for maximizing own payoffs. In the proposed Bayesian game, a strategy for Agent 1 can be represented by a function $a_1(t_1)$ specifying the market ratio that Agent 1 would choose. In a Bayesian Nash equilibrium [23], Agent 1’s strategy $a_1(t_1)$ is a best response to Agent 2’s strategy $a_2(t_2)$, and vice versa. Based on Definition of Bayesian Nash Equilibrium [23], The pair of strategies $(a_1(t_1), a_2(t_2))$ is a Bayesian Nash Equilibrium, if for each $t_y$ in $[0,1]$, ($y=1,2$), $a_y(t_y)$ solves:

$$
\text{max}_{a_1, a_2} \sum_{t_1, a_1 \in t_1} u_y(a_1^*(t_1), a_2^*(t_2); t)b_y(t_{-y} | t_y)
$$

For a given value of $a_1$, the best response of Agent 1 is obtained by:

$$
\text{max}_{a_1} \frac{(y + c)}{2} \Pr(t_1 \geq t_2)
$$

where $c$ is the expected strategy cost and is calculated by:

Figure 6.8: The Proposed Bayesian Game for Strategic Platform Design
where \( i \) is the member of design strategies including EP modules, \( i=1,2,\ldots,n \). \( v \) is the market price obtained by:

\[
v = 2 \times w_i c(s_i) \times t_2
\]

(6.27)

and the probability of \( t_1 \geq t_2 \) is

\[
\Pr(t_1 \geq t_2) = 1 - \Pr(t_1 < t_2) = 1 - \int_0^1 F_{T_1}(t_2) f_{T_2}(t_2) dt_2
\]

(6.28)

where, \( F_{T_1}(t_2) = \int_0^{t_2} f_{T_1}(t_1) dt_1 \). Here, \( f_{T_1}(t_1) \) and \( f_{T_2}(t_2) \) are a uniform distribution over \([0,1]\).

For Agent 2, we can obtain the best response through the same processes. In this game, strategies for Agent 1 represent the various module design methods depending on EP modules in a product family. Therefore, engineering parameters’ values for platform design can be determined by selecting strategies in uncertain e-market environments. In the next section, the proposed method is applied to a case study involving a family of power tools.

### 6.4.4 Case Study: Power Tool Family

To demonstrate implementation of the proposed Bayesian game, the power tool family is investigated. The case study in Section 6.3.3 was focused on determining a platform design strategy based on module design costs for the power tool family in a dynamic market environment. Meanwhile, in this section, market share ratios and
customers’ preferences are considered as uncertain information to determine a platform design strategy for the power tool family.

**Determine Strategy for Platform Design**

Common and unique module design can be determined from an auction based on market mechanisms to minimize total production cost (see Section 6.3.3). Since the motor module and the battery module are EP modules, we suppose that they are involved in designing the platform. Therefore, appropriate EP values for EP modules can be determined by a game to design the platform. The designer can configure components to develop design strategies based on the EP values for the game with a customer. Table 6.11 shows possible design strategies of the motor module and the battery module for a platform. For example, from Table 6.11, a design strategy for the motor module consists of three components: Motor A, Gear A, and Shaft A. The resulting torque is 7.5 N.m. (torque 3 N.m × gear ratio 2.5), which satisfies the target engineering parameter for the sander.

<table>
<thead>
<tr>
<th>Module</th>
<th>Strategy</th>
<th>Component</th>
<th>EP level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motor</td>
<td>$s_{m1}$</td>
<td>A,A,A</td>
<td>7.5 N.m.</td>
</tr>
<tr>
<td></td>
<td>$s_{m2}$</td>
<td>A,B,A</td>
<td>12 N.m.</td>
</tr>
<tr>
<td></td>
<td>$s_{m3}$</td>
<td>B,A,A</td>
<td>12.5 N.m.</td>
</tr>
<tr>
<td></td>
<td>$s_{m4}$</td>
<td>B,B,A</td>
<td>20 N.m.</td>
</tr>
<tr>
<td>Battery</td>
<td>$s_{b1}$</td>
<td>A</td>
<td>5.0 A</td>
</tr>
<tr>
<td></td>
<td>$s_{b2}$</td>
<td>B</td>
<td>7.0 A</td>
</tr>
</tbody>
</table>
To determine the expected strategy cost, we use the expected cost functions, Eq. (6.18). The strategies also should satisfy the constraints when the strategies are selected. The EP costs can be calculated by the EP cost model given by Eq. (6.1). We assume that the loss cost of EP is defined as an additional component cost, which is 1 unit and 2 units for torque and current, respectively. We assume that a factor of overhead is 2 units. The designer calculates the excepted strategy cost based on the minimum cost of each design method for each product. For example, $s_{m1}$ has a module cost of 13 ($=5+4+4$) and an assembly cost of 4 ($=2+1+1$) for satisfying the constraint of the sander. To satisfy the EP value constraints for the circular saw, the jig saw, the drill, and the brad nailer, additional costs are 5, 5, 8, and 8 units, respectively. Therefore, the expected strategy cost for $s_{m1}$ is 44.4, if the value of market share ratio is 1. For a customer, the market price’s weight coefficients of four strategies can be determined by the proportion of the maximum numbers of satisfying EP value constraints mentioned in Section 6.4.3. Table 6.12 shows four expected strategy costs and the weights of the market price coefficients, when the values of market share ratio (MSR) and customers’ preference (CP) are 1, respectively.

Table 6.12: Expected Strategy Costs and Market Prices (MSR and CP =1)

<table>
<thead>
<tr>
<th>EP module</th>
<th>Strategy</th>
<th>Expected strategy cost</th>
<th>Market price coefficient</th>
<th>Market price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motor</td>
<td>$s_{m1}$</td>
<td>44.4</td>
<td>0.2</td>
<td>17.76</td>
</tr>
<tr>
<td></td>
<td>$s_{m2}$</td>
<td>40.4</td>
<td>0.6</td>
<td>48.48</td>
</tr>
<tr>
<td></td>
<td>$s_{m3}$</td>
<td>40</td>
<td>0.6</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>$s_{m4}$</td>
<td>42</td>
<td>1</td>
<td>84</td>
</tr>
<tr>
<td>Battery</td>
<td>$s_{b1}$</td>
<td>13.2</td>
<td>0.6</td>
<td>15.84</td>
</tr>
<tr>
<td></td>
<td>$s_{b2}$</td>
<td>14</td>
<td>1</td>
<td>28</td>
</tr>
</tbody>
</table>
Analysis for a Bayesian Game in Platform Design

The game between a designer and a customer for platform design of this family is defined as the proposed Bayesian game described in Section 6.4.3. Table 6.13 shows the Bayesian game for determining EP modules with three agents. In this case study, the Bayesian game focuses on determining an EP module for a platform based on designer’s action.

Table 6.13: A Bayesian Game for Determining a Design Strategy

<table>
<thead>
<tr>
<th>Bayesian game</th>
<th>Agent 1</th>
<th>Agent 2</th>
<th>Agent 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Player</td>
<td>Motor module designer</td>
<td>Battery module designer</td>
<td>Customer</td>
</tr>
<tr>
<td>Action</td>
<td>Select a design strategy</td>
<td>Select a design strategy</td>
<td>Select a market price</td>
</tr>
<tr>
<td>Type</td>
<td>Market share ratio</td>
<td>Market share ratio</td>
<td>Customer’s preference</td>
</tr>
<tr>
<td>Belief</td>
<td>Uniform probability [0, 1]</td>
<td>Uniform probability [0, 1]</td>
<td>Uniform probability [0, 1]</td>
</tr>
<tr>
<td>Payoff function</td>
<td>Profit ((v+c)/2- c)</td>
<td>Profit ((v+c)/2- c)</td>
<td>Profit ((c+v)/2- v)</td>
</tr>
</tbody>
</table>

To determine the best response of Agent 1 and Agent 2, we performed a sensitivity analysis for various market share ratios based on customer’s strategies mentioned in Section 6.3.4. Figures 6.9 and 6.10 show the two designers’ payoffs for platform strategies based on different market price strategies, when the customer’s preference is 0.6. In these cases, \(s_{m3}\) and \(s_{b1}\) are dominated strategies based on the designer’s payoff. Therefore, a motor module with torque 3 N.m. and gear ratio 2.5 and a battery module with current 5.0 A can be designed as a new platform. The results from a sensitivity analysis provide a designer with information for determining a platform strategy in an uncertain market environment. In conclusion, if the customer’s preference is predicted as 0.6, a new platform for five tools can consist of three modules that include an electronic module, a motor module (torque 3 N.m., gear ratio 2.5), and a battery module (current 5.0 A).
Through the case study, the proposed Bayesian game was demonstrated to determine the EP-value of an EP module to select appropriate modules for the platform. Therefore, the Bayesian game can facilitate product family design in various dynamic market environments.
6.5 Extension to Service Family Design

In this section, concepts from platform-based product family design are extended to module-based service family design. The objective in this section is to apply the proposed game theoretical approach as mentioned in Section 6.4.3 to determine a platform strategy for service family design in uncertain market environments. More detail of determine expected service module cost is described in next section.

6.5.1 Cost of Expected Strategies

A functional service module consists of service process modules that are represented by service components to perform a service in a service family. Let $C^F_j$ be functional module cost of functional module $j$, $C^P_{kj}$ process module cost of process module $k$ in function $j$, $C^C_{kli}$ process component cost of process component $l$ in process module $k$, $C^A_{klt}$ attribute cost of attribute $t$ in the process component $l$ of process module $k$, and $C^I_{jk}$ interface cost of process module $k$ in function $j$. Interface cost depends on the number of service process modules. The functional module cost for function $j$ is proportional to the number of service process modules as follows:

$$C^F_j \propto \sum_{k,j} (C^P_{jk} + C^I_{jk})$$  \hspace{1cm} (6.30)

Process module cost, $C^P_{jk}$, is depended on the number of process components and can be represented by:
Process component cost, \( C_{kl}^C \), is proportional to the number of component’s attributes and can be obtained by:

\[
C_{kl}^C \propto \sum_{t \in \propto} C_{kl}^C
\]  

(6.31)

By introducing coefficients, \( \alpha, \beta, \) and \( \lambda \), the functional module cost can be formulated at the operational level as follows:

\[
C_j^F = \alpha \sum_{k \in j} (\beta \sum_{l \in k} \lambda \sum_{t \in l} C_{klt}^A + C_{jk}^I)
\]  

(6.33)

where \( \alpha, \beta, \) and \( \lambda \), are mapping cost coefficients related to process modules, process components, and attributes, respectively (\( 0 < \alpha, \beta, \lambda < \infty \)). To set the functional module cost, an industrial case study is necessary to determine the mapping cost coefficients by investigating the relationships between modules in various conditions [17].

Based on functional service modules, a platform designer determines a feasible set of strategies for the service platform based on his/her design knowledge. The strategies are represented as alternative design methods for a service platform and can be constructed by combining process modules based on common functions in a service family. Let \( G \) be a set of strategies, \( Y \) a set of functions, \( S \) a set of services in a service family, and \( Q \) a set of module quantities. \( G, Y, S, \) and \( Q \) are finite sets.

The expected strategy cost, \( c(g_i) \), for designer’s strategy \( g_i \) \( (i=1,\ldots,N) \) is estimated by an expected cost function: \( f^i : G \times Y \times S \times Q \rightarrow \mathbb{R} \). Hence, the real number of \( f^i(g,y,s,q) \) represents the cost of strategy paying \( y \) at quantity \( q \) for performing
strategy $g$ for a platform in $s$ services. For example, the expected strategy cost for $s$ can be determined based on functional costs as:

$$f^i(g, y, s, q) = \eta \times \frac{\sum_{i \in G} C_i}{l \times q}$$

(6.34)

where $\eta$ is a factor for overhead, and $l$ is a strategy weighting function as follows:

$$l = \begin{cases} 1, & \text{if module is unique} \\ \chi, & \text{otherwise} \end{cases}$$

(6.35)

where $\chi$ is the number of services including $g_i$ in a service family, and $q$ is a volume discount factor or market share ratio that is considered as a quantity function. For a given set of services, $c(g_i)$ varies depending on choosing a strategy for platform design. The next section discusses a Bayesian game model for determining a platform design strategy. In the next section, the proposed Bayesian game as mentioned in Section 6.4.3 is applied to determine a process module for platform design using a case study involving the family of banking services.

### 6.5.2 Case Study: Checking Account Services

To demonstrate the implementation of the proposed Bayesian game, the family of banking services from Section 4.3.4 is used. This case study focuses on a process-based platform for the family of banking services at the conceptual stage of development.

From the information in the FPM in Table 4.5, we can define four design strategies based on variant functional modules and service process modules as shown in Table 6.15. Seven alternatives for platform design using three variant functional modules
can be considered as four design strategies with respect to process modules. For example, 
D1 and D5 in Table 6.14 can be designed by the same process modules to perform their 
functional modules, F10 and F12.

Table 6.14: Platform Strategies Based on Service Process Modules

<table>
<thead>
<tr>
<th>Design</th>
<th>Functional module</th>
<th>Process module</th>
<th>Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>F10</td>
<td>P4, P6, P8, P10</td>
<td>G1</td>
</tr>
<tr>
<td>D2</td>
<td>F11</td>
<td>P2, P5, P6, P7, P8, P10</td>
<td>G2</td>
</tr>
<tr>
<td>D3</td>
<td>F12</td>
<td>P6, P8, P10</td>
<td>G3</td>
</tr>
<tr>
<td>D4</td>
<td>F10, F11</td>
<td>P2, P4, P5, P6, P7, P8, P10</td>
<td>G4</td>
</tr>
<tr>
<td>D5</td>
<td>F10, F12</td>
<td>P4, P6, P8, P10</td>
<td>G1</td>
</tr>
<tr>
<td>D6</td>
<td>F11, F12</td>
<td>P2, P5, P6, P7, P8, P10</td>
<td>G2</td>
</tr>
<tr>
<td>D7</td>
<td>F10, F11, F12</td>
<td>P2, P4, P5, P6, P7, P8, P10</td>
<td>G4</td>
</tr>
</tbody>
</table>

In this section, we assumed that the cost of a service component’s attributes and 
the interface cost of a process module are 1 unit, respectively. Suppose \( \alpha = \beta = \gamma = 1 \), 
and then, the functional cost for strategy can be calculated by Eq. (6.34). We assumed 
that a factor of overhead is 2 units. For example, strategy G3 has a component cost of 29 
\((=12+8+9)\) and an interface cost of 3 units according to its process modules. Therefore, 
the expected strategy cost for S3 is 21.33, if the value of market share ratio is 1. For a 
customer, the market price’s weight coefficients of four strategies can be determined by 
the proportion of the maximum numbers of providing services for four service plans as 
mentioned in Section 6.3.3. Table 6.15 show the cost for four expected strategies and the 
weights of the market price coefficients when the values of market share ratio (MSR) and 
customers’ preference (CP) are 1.
The game between a service designer and a customer for platform design of this service family is defined as the proposed Bayesian game that is described in Section 6.4.3. Table 6.16 summarizes the Bayesian game for determining modules with two agents. In this case study, the Bayesian game focuses on determining a functional module for a service platform based on designer’s action.

To determine the best response of Agent 1, we performed a sensitivity analysis for various market share ratios based on customer’s preferences as mentioned in Section 6.3.3. Figures 6.11 and 6.12 show the designer’s payoffs for platform strategies based on different market price strategies, when the customer’s preference is 0.4 and 0.5 respectively. For example, four design strategies in Figure 6.11 have no profit if the market share ratio is less than 0.7 with respect to different market prices. Otherwise, G1, G2, and G3 in Figure 6.12 have profits as increasing the value of market share ratios if market price strategies are G1 and G2. In these cases, G3 is a dominated strategy based
on the designer’s payoff. Therefore, F12 can be designed as a new service platform with common service process modules. The results from a sensitivity analysis provide a designer with information for determining a service platform in an uncertain market environment. In conclusion, if the customer’s preference is predicted as 0.5, a new service platform for four checking account services can consist of 10 functional modules that are 9 common modules and one variant module for Maintenance fee.

Figure 6.11: Payoffs of Agent 1 when Customer’s Preference is 0.4
Through the case study, we demonstrated that the proposed Bayesian game can be used to determine functional modules in selecting appropriate service process modules for the platform. We believe that the Bayesian game can facilitate service family design in various dynamic market environments.

6.6 Summary and Preview

This chapter introduced a new mathematical model and a dynamic multi-agent system (DMAS) based on a market-based negotiation mechanism for Step 4 in the SMPDM. In the DMAS, agents play an important role to perform product design using a negotiation mechanism. An agent environment is modeled as an e-market that consists of economically-motivated agents to explain agents’ behaviors and roles. In the DMAS, a negotiation mechanism was used to solve a task allocation problem that was considered...
as a module selection problem. A negotiation process was demonstrated to determine a platform using a case study involving a family of power tools.

To investigate strategic platform design between products in an uncertain market environment, a game theoretic approach was described. A module selection problem was considered as a strategic game with incomplete information that was represented by products’ market share ratio and customer’s preference. A Bayesian game was employed to model uncertainty situations regarding market environments and decided strategic equilibrium solutions for selecting module design strategy based on the expected strategy cost in product family design. The proposed Bayesian game has been applied to determine platform design strategies using case studies involving the families of power tools and banking services.

The next chapter provides implementation of the SMPDM using two case studies involving a power tool family and a checking account service family.
Chapter 7
Implementing the SMPDM

This chapter presents two examples to demonstrate the usefulness and applicability of the SMPDM for determining a platform design strategy in dynamic and uncertain e-market environments. The SMPDM is employed step-by-step for two case studies involving a power tool family and checking account services to provide proof of concept that it works in Sections 7.1 and 7.2, respectively. A summary of this chapter and a preview of the next are given in Section 7.3.

7.1 Example 1: Power Tool Family

In this example, five tools from the power tool family are investigated to demonstrate the SMPDM (see Table 5.1). Current common components for the products consist of a battery and a battery terminal connector as shown in Appendix A. Through the example, a new platform is proposed to increase commonality in the products based on platform design strategies in a dynamic market environment.

7.1.1 Step 1: Analyze Design Data

The first step in the SMPDM is to analyze design data to support design knowledge representation. Products and functional requirements are defined in this step
to establish design strategies. For example, Table 7.1 provides functional design information for a circular saw by a functional model (refer to Appendix A).

Table 7.1: Customer Needs and Functional Requirements for Circular Saw

<table>
<thead>
<tr>
<th>Product</th>
<th>Customer Needs</th>
<th>Module</th>
<th>Function</th>
<th>Functional Requirements</th>
</tr>
</thead>
</table>
| Circular saw | - Circle cutting  
               - Electric method  
               - Lightweight and portable  
               - Safety  
               - Varying speed  
               - Variable cut angle | Electronic | import | - Torque: 12 N.m.  
                          actuate | transfer | - Battery 5.0 A |
|            |                | Motor  | convert | store                   |
|            |                | Battery | export | store                   |
|            |                | Blade  | import  | export                  |
|            |                | Input  | import  | store                   |
|            |                | Gear train | import | transfer               |

7.1.2 Step 2: Knowledge Representation for the Power Tool Family

Design knowledge for the power tool family is developed in TCO and provides information related to modules and components in the power tools. Attributes for describing a component are used to perform the module identification method as described in Chapter 5. Table 7.2 shows product representation for a circular saw using the coding approach mentioned in Section 5.2 (see Appendix B).
7.1.3 Step 3: Identify Modules and Strategies

Using the module identification method described in Chapter 5, modules for the products were identified as either unique, variant, or common modules as summarized in Table 7.3. Since the motor and the battery modules have EP values in common modules, decision-making for these modules is needed to determine EP-values for developing a platform. Therefore, in this case, platform strategies can be developed through combinations of the EP modules and the common modules listed in Table 7.3. Meanwhile, the variant modules and the unique modules are used for customized design strategies to increase customer satisfaction.

Table 7.2: Product Representation for Circular Saw

<table>
<thead>
<tr>
<th>Product</th>
<th>Function</th>
<th>Module</th>
<th>Attribute codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1,1</td>
<td>Electronic</td>
<td>import</td>
<td>Attribute codes</td>
</tr>
<tr>
<td>x1,2</td>
<td>Motor</td>
<td>transfer</td>
<td>Attribute codes</td>
</tr>
<tr>
<td>x1,3</td>
<td>Battery</td>
<td>transfer</td>
<td>Attribute codes</td>
</tr>
<tr>
<td>x1,4</td>
<td>Blade</td>
<td>store</td>
<td>Attribute codes</td>
</tr>
<tr>
<td>x1,5</td>
<td>Input</td>
<td>import</td>
<td>Attribute codes</td>
</tr>
<tr>
<td>x1,6</td>
<td>Gear train</td>
<td>transfer</td>
<td>Attribute codes</td>
</tr>
</tbody>
</table>
7.1.4 Step 4: Specify Platform Strategy

Design information related to the EP modules is determined by TCO as described in Chapter 4. In dynamic e-market environments, suppose that the component specifications for the EP modules are obtained by the auctions described in Section 6.3. Table 7.4 shows the combinations of components in the EP modules to develop design strategies based on their specifications.

Table 7.4: Module Categorization and Design Strategies for the Power Tool Family

<table>
<thead>
<tr>
<th>Categorization</th>
<th>Module (EP)</th>
<th>Design strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common</td>
<td>• Electronic module</td>
<td>Platform design</td>
</tr>
<tr>
<td></td>
<td>• Battery module (torque)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Motor module (current)</td>
<td></td>
</tr>
<tr>
<td>Variant</td>
<td>• Input module</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Gear train module</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Conversion module</td>
<td></td>
</tr>
<tr>
<td>Unique</td>
<td>• Blade module</td>
<td>Customized products</td>
</tr>
<tr>
<td></td>
<td>• Sander mounting module</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Magazine module</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Bit module</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Gear train module</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Conversion module</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Sander mounting module</td>
<td></td>
</tr>
</tbody>
</table>

Table 7.3: Module Categorization and Design Strategies for the Power Tool Family

<table>
<thead>
<tr>
<th>Categorization</th>
<th>Module (EP)</th>
<th>Design strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common</td>
<td>• Electronic module</td>
<td>Platform design</td>
</tr>
<tr>
<td></td>
<td>• Battery module (torque)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Motor module (current)</td>
<td></td>
</tr>
<tr>
<td>Variant</td>
<td>• Input module</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Gear train module</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Conversion module</td>
<td></td>
</tr>
<tr>
<td>Unique</td>
<td>• Blade module</td>
<td>Customized products</td>
</tr>
<tr>
<td></td>
<td>• Sander mounting module</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Magazine module</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Bit module</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Gear train module</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Conversion module</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Sander mounting module</td>
<td></td>
</tr>
</tbody>
</table>
Table 7.5 shows the results of two agent-based decision-making methods, a negotiation mechanism and a Bayesian game as described in Sections 6.3 and 6.4, respectively. For the Motor module, a design strategy (torque 3 N.m. × gear ratio 2.5) is a dominated strategy in the two decision-making methods. Meanwhile, the dominated design strategy for the Battery module is dependent on market situations that are represented by the degree of design information as shown in Table 7.5. Comparing this to the current platform for the five products, we can increase the number of common modules based on common functional features. This represents a 23% increase in the number of common modules within the family.

Table 7.4: EP Modules’ Components and Design Strategies

<table>
<thead>
<tr>
<th>Module</th>
<th>Component</th>
<th>Specification</th>
<th>Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motor</td>
<td>Motor A</td>
<td>Torque 3 N.m.</td>
<td>$S_{m1}$: Motor A, Gear A, Shaft A</td>
</tr>
<tr>
<td></td>
<td>Motor B</td>
<td>Torque 5 N.m.</td>
<td>$S_{m2}$: Motor A, Gear B, Shaft A</td>
</tr>
<tr>
<td>Gear</td>
<td>Gear A</td>
<td>Gear ratio 2.5</td>
<td>$S_{m3}$: Motor B, Gear A, Shaft A</td>
</tr>
<tr>
<td></td>
<td>Gear B</td>
<td>Gear ratio 4.0</td>
<td>$S_{m4}$: Motor B, Gear B, Shaft A</td>
</tr>
<tr>
<td>Shaft</td>
<td>A</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Battery</td>
<td>Battery A</td>
<td>Current 5.0 A</td>
<td>$S_{b1}$: Battery A</td>
</tr>
<tr>
<td></td>
<td>Battery B</td>
<td>Current 7.0 A</td>
<td></td>
</tr>
</tbody>
</table>

Table 7.5: Platform Design Strategy for EP Modules

<table>
<thead>
<tr>
<th>Decision-making</th>
<th>Module</th>
<th>Dominated strategy</th>
<th>Market situation</th>
</tr>
</thead>
<tbody>
<tr>
<td>A negotiation mechanism</td>
<td>Motor</td>
<td>Motor B, Gear A, Shaft A</td>
<td>Complete information</td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td>Battery B</td>
<td></td>
</tr>
<tr>
<td>A Bayesian game</td>
<td>Motor</td>
<td>Motor B, Gear A, Shaft A</td>
<td>Incomplete information</td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td>Battery A</td>
<td></td>
</tr>
</tbody>
</table>
Through the case study, the SMPDM was demonstrated to determine the EP-value for EP module to select appropriate modules for the platform. Therefore, the SMPDM can facilitate product family design in dynamic and uncertain market environments.

7.2 Example 2: Checking Account Service Family

In this example, a family of a checking account services (see Table 4.5) is investigated to demonstrate the SMPDM for designing a family of services. Current variant functional options for the services are Trade stock online, Optional business economic checking, and Maintenance fee. Through the example, a new platform is proposed to increase functional options based on service processes under incomplete information.

7.2.1 Step 1: Analyze Design Data

The first step in the SMPDM is to analyze the services to support design knowledge representation using object-oriented concepts. To establish design strategies for the services, the relationship between functional modules and process modules are defined by a function-process matrix (FPM) in this step (refer to Chapter 4).

7.2.2 Step 2: Knowledge Representation for the Tool Family

Design knowledge for the services is developed from the service ontology and provides information related to their modules and components. Attributes for describing a
component are used to perform the module identification method as described in Chapter 5. Table 7.6 shows the service representation for Plan A using the coding approach mentioned in Section 5.2 (refer to Table 5.7).

Table 7.6: Service Process Representation for Plan A

<table>
<thead>
<tr>
<th>Plan</th>
<th>Module Process</th>
<th>Component</th>
<th>Attributes</th>
<th>Activity</th>
<th>Object</th>
<th>Input No.</th>
<th>Object</th>
<th>Input flow</th>
<th>Output flow</th>
<th>Status</th>
<th>Activity</th>
<th>Object</th>
<th>Input flow</th>
<th>Output flow</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Saving</td>
<td>Request</td>
<td>Customer</td>
<td>Account No.</td>
<td>Money</td>
<td>-</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>Request</td>
<td>Customer</td>
<td>Account No.</td>
<td>Money</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Accept</td>
<td>Employee</td>
<td>Money</td>
<td>Amount</td>
<td>Balance</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>0</td>
<td>Accept</td>
<td>Employee</td>
<td>Money</td>
<td>Amount</td>
<td>Balance</td>
</tr>
<tr>
<td></td>
<td>Inform</td>
<td>Employee</td>
<td>Amount</td>
<td>Amount</td>
<td>-</td>
<td>3</td>
<td>2</td>
<td>6</td>
<td>0</td>
<td>Inform</td>
<td>Employee</td>
<td>Amount</td>
<td>Amount</td>
<td>-</td>
<td>3</td>
</tr>
<tr>
<td>Drawing</td>
<td>Request</td>
<td>Customer</td>
<td>Account No.</td>
<td>Amount</td>
<td>-</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>Request</td>
<td>Customer</td>
<td>Account No.</td>
<td>Amount</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Accept</td>
<td>Employee</td>
<td>Amount</td>
<td>Money</td>
<td>Balance</td>
<td>1</td>
<td>2</td>
<td>6</td>
<td>4</td>
<td>0</td>
<td>Accept</td>
<td>Employee</td>
<td>Amount</td>
<td>Money</td>
<td>Balance</td>
</tr>
<tr>
<td></td>
<td>Inform</td>
<td>Employee</td>
<td>Amount</td>
<td>Amount</td>
<td>-</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>0</td>
<td>Inform</td>
<td>Employee</td>
<td>Amount</td>
<td>Amount</td>
<td>-</td>
</tr>
<tr>
<td>Transferring</td>
<td>Confirm</td>
<td>Database</td>
<td>Amount</td>
<td>Amount</td>
<td>-</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>2</td>
<td>Confirm</td>
<td>Database</td>
<td>Amount</td>
<td>Amount</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Inform</td>
<td>Employee</td>
<td>Amount</td>
<td>Message</td>
<td>-</td>
<td>3</td>
<td>2</td>
<td>6</td>
<td>8</td>
<td>0</td>
<td>Inform</td>
<td>Employee</td>
<td>Amount</td>
<td>Message</td>
<td>-</td>
</tr>
<tr>
<td>Trading Stocks</td>
<td>Request</td>
<td>Customer</td>
<td>Customer ID</td>
<td>Account No.</td>
<td>Message</td>
<td>-</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>8</td>
<td>0</td>
<td>Request</td>
<td>Customer</td>
<td>Customer ID</td>
<td>Account No.</td>
</tr>
<tr>
<td></td>
<td>Query</td>
<td>Employee</td>
<td>Message</td>
<td>Message</td>
<td>-</td>
<td>4</td>
<td>2</td>
<td>8</td>
<td>8</td>
<td>0</td>
<td>Query</td>
<td>Employee</td>
<td>Message</td>
<td>Message</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Inform</td>
<td>Employee</td>
<td>Message</td>
<td>Message</td>
<td>-</td>
<td>4</td>
<td>2</td>
<td>8</td>
<td>8</td>
<td>0</td>
<td>Inform</td>
<td>Employee</td>
<td>Message</td>
<td>Message</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Conform</td>
<td>Database</td>
<td>Customer ID</td>
<td>Customer ID</td>
<td>-</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Conform</td>
<td>Database</td>
<td>Customer ID</td>
<td>Customer ID</td>
<td>-</td>
</tr>
<tr>
<td>Check writing</td>
<td>Accept</td>
<td>Database</td>
<td>Customer ID</td>
<td>Account No.</td>
<td>-</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>Accept</td>
<td>Database</td>
<td>Customer ID</td>
<td>Account No.</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Reject</td>
<td>Database</td>
<td>Customer ID</td>
<td>Message</td>
<td>-</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>8</td>
<td>0</td>
<td>Reject</td>
<td>Database</td>
<td>Customer ID</td>
<td>Message</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Inform</td>
<td>Database</td>
<td>Customer ID</td>
<td>Credit</td>
<td>-</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>Inform</td>
<td>Database</td>
<td>Customer ID</td>
<td>Credit</td>
<td>-</td>
</tr>
<tr>
<td>ID certifying</td>
<td>Confirm</td>
<td>Database</td>
<td>Amount</td>
<td>Balance</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>2</td>
<td>Confirm</td>
<td>Database</td>
<td>Amount</td>
<td>Balance</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Inform</td>
<td>Database</td>
<td>Amount</td>
<td>Balance</td>
<td>-</td>
<td>3</td>
<td>3</td>
<td>6</td>
<td>7</td>
<td>0</td>
<td>Inform</td>
<td>Database</td>
<td>Amount</td>
<td>Balance</td>
<td>-</td>
</tr>
</tbody>
</table>

7.2.3 Step 3: Identify Modules and Strategies

Using the module identification method described in Section 5.1, process modules for the services are identified as unique, variant, and common process modules based on their activities as shown in Table 7.7. Since the variant functional modules have the process modules including Accept, Inform, Request, and Query activities, a new platform can be designed by integrating the common functions and variant functions to increase and improve customer satisfaction. In this case, platform strategies are developed by combinations of the variant functional modules based on their process modules.
7.2.4 Step 4: Specify Platform Strategy

Design information related to process modules and service components is determined from the service ontology mentioned in Chapter 4. Platform design strategies for the services were developed by combinations of functional modules based on their process modules as shown in Table 7.8. Using the proposed Bayesian game in Section 6.4, a design strategy including the Maintenance fee function dominated based on the designer’s payoff, if the customer’s preference is predicted as 0.5. In this case, a new service platform for four checking account services can consist of 10 functional modules that are 9 common modules and one variant module for Maintenance fee. Comparing this to the current platform for the four checking account services, we can increase the number of common options based on common service processes. This represents a 6% increase in the number of common options within the family.

Table 7.7: Module Categorization and Design Strategies for the Checking Account Services

<table>
<thead>
<tr>
<th>Categorization</th>
<th>Activity</th>
<th>Design strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common</td>
<td>Accept, Inform, Request, Query</td>
<td>Platform design</td>
</tr>
<tr>
<td>Variant or Unique</td>
<td>Confirm, Reject, Proposal</td>
<td>Customized design</td>
</tr>
</tbody>
</table>
Through the case study, we demonstrated that the SMPDM can be used to determine functional modules in selecting appropriate service process modules for the platform. We believe that the SMPDM can facilitate service family design in dynamic and uncertain market environments.

7.3 Summary and Preview

In this chapter, the SMPDM was presented step-by-step to demonstrate proof-of-concept and was implemented using two industrial case studies, a power tool family and a checking account service family. Through these two examples, the SMPDM was successfully used to determine a platform design strategy for a product or service family in various market situations. Therefore, the SMPDM can provide an appropriate method to support decision-making for product and service family development in dynamic market environments. The next chapter provides the summary, contributions, and recommendations from this research along with a discussion of the research limitations and future work.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Process module</th>
<th>Functional module</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>P4, P6, P8, P10</td>
<td>Trade stock online, Maintenance fee</td>
</tr>
<tr>
<td>S2</td>
<td>P2, P5, P6, P7, P8, P10</td>
<td>Optional business economic checking, Maintenance fee</td>
</tr>
<tr>
<td>S3*</td>
<td>P6, P8, P10</td>
<td>Maintenance fee</td>
</tr>
<tr>
<td>S4</td>
<td>P2, P4, P5, P6, P7, P8, P10</td>
<td>Trade stock online, Optional business economic checking, Maintenance fee</td>
</tr>
</tbody>
</table>

*: A dominated strategy, if the customer’s preference is predicted as 0.5
Chapter 8
Contributions and Recommendations

In this dissertation, the Strategic Module-based Platform Design Method (SMPDM) for determining a platform design strategy has been developed, presented, and tested to support product and service family design in dynamic and uncertain market environments. Section 8.1 reviews the research objectives and the proposed method. The resulting contributions and limitations of the research are summarized in Section 8.2. In Section 8.3, future work is discussed, and concluding remarks are given in Section 8.4.

8.1 Research Summary

As stated in the introduction of Chapter 1, the primary objective in this dissertation is to develop the Strategic Module-based Platform Design Method (SMPDM) for determining a platform strategy to support product and service family design in dynamic and uncertain market environments. Engineering design, operation research, artificial intelligent, applied statistics, and strategic management methods and tools are integrated to form the Strategic Module-based Platform Design Method. In addition, the concept of designing a service family is introduced and formalized in the dissertation. Secondary objectives were identified in order to achieve the primary objective in the dissertation. Their elaboration and verification have provided the context in which the research has proceeded.
The primary objective posed for this work is embodied by the Strategic Module-based Platform Design Method: a Method that manages and determines Strategic Design for a Module-based Platform which can be applied to appropriate families of products or services in dynamic and uncertain market environments. The method consists of managing design information and knowledge, modeling a strategic problem, and formulating and solving the problem. Case studies involving power tools and banking services were used to demonstrate how the method manages platform design strategies that are represented by alternative design methods in dynamic and uncertain market environment. To determine an optimal platform design strategy, a negotiation mechanism was applied to the family of power tools, and the proposed Bayesian game was applied to two case studies. The method is generally applicable to other examples in this class of problems: modular product and service platforms whose design knowledge can be represented and modeled with respect to cost, reflecting dynamic and uncertain market information.

In this research, an ontology is investigated to represent knowledge for sharing and reusing design information related to products and services. To represent products and components, the Techspecs Concept Ontology (TCO) was utilized and demonstrated using a case study involving a family of power tools. TCO provides functional representation-based semantics of products or components to better reflect customers’ preferences and market needs. In addition, assembly relationship information and technical specifications in TCO help designers search all related components based on functional requirements and the feasibility of assembly design. For the service ontology, a function-process matrix (FPM) was used to identify relationships between service
functions and service processes in a family of services. Based on a graph model and object-oriented concepts, a service process model was introduced to describe a service represented by the service ontology. The service ontology was utilized to represent service design knowledge within a case study involving a family of four banking services.

Data mining techniques were applied to develop a method for identifying a platform along with variant and unique modules by clustering based on the similarity of functional features in a family of products and services. Fuzzy c-means clustering was employed to cluster the functional features of products and services based on their similarity. The resulting clusters yielded the initial modules, which were then classified using fuzzy set theory to determine the platform within the family. Case studies for product and service families have demonstrated to work the proposed method for determining a new platform. The results of comparison between the current platform and the proposed platform are described in Sections 5.2.4 and 5.3.2, respectively. The proposed method can provide designers with a module-based platform and modules that can be adapted to product and service design during conceptual design.

To determine a platform for families of products and services in a dynamic and uncertain electronic market environment, a market-based negotiation mechanism, a Bayesian game, and a learning mechanism were investigated based on module-based platform concept and a mathematical model as introduced in Section 6.1. Sections 6.2 and 6.3 outline a method for designing a dynamic multi-agent system (DMAS) and developing a market-based negotiation mechanism for product platform design. In the DMAS, a negotiation mechanism was used to solve a module selection problem. A case
study involving a family of power tools was used to demonstrate the negotiation process to determine a platform in a dynamic market environment. To investigate strategic platform design for families of products and services in an uncertain market environment, a game theoretic approach was proposed. A module selection problem was considered as a strategic game with incomplete information that was represented by the market share ratio and customer’s preference. A Bayesian game was employed to model uncertainty situations regarding market environments and identify strategic equilibrium solutions for selecting module design strategies based on the expected cost. The proposed Bayesian game has been applied to determine platform design strategies using the same two case studies involving the family of power tools and the family of banking services.

8.2 Contributions and Limitations of the Research

The contributions offered in this dissertation are first introduced in Section 1.3.2 and then realized throughout the dissertation. The primary contribution from this work is embodied in the Strategic Module-based Platform Design Method (SMPDM), which is a method to identify, manage, and determine a platform design strategy for families of products and services in dynamic and uncertain market environments. The other contributions can be summarized as follows:

Contributions Related to the SMPDM:

- Techsepcs Concept Ontology for product representation, see Section 4.2.
- Service ontology for service representation, see Section 4.3.
• A method for identifying modules and a platform in a product and service family, see Section 5.1
• A mathematical model for module-based platform design in a product and service family, see Sections 6.1 and 6.5.1.
• A dynamic multi-agent system for representing dynamic product design environments based on market mechanisms, see Section 6.2.
• A negotiation mechanism based on a market mechanism for agent-based decision-making, see Section 6.3
• A game theoretic approach for determining a design strategy using a Bayesian game for a product and service family, see Sections 6.4 and 6.5.2

Contributions Related to Product Family Design:
• Investigation for a module-based approach to support product family design, see Chapters 4 and 5.
• A method for identifying modules and a platform using data mining techniques in a conceptual design phase, see Chapter 5.
• Market-based design decision-making for product family design in dynamic and uncertain design environments see Chapter 6.

Contributions Related to Service Design:
• Extension the concepts from product platform design to service platform design for a family of services, see Sections 2.7.
• A service process model to model and represent services using modular design concepts and object-oriented concepts, see Section 4.3.2.

The contributions made in the area of product and service family design, specifically the method of determining strategic platform design, represent an addition to the fundamental knowledge of the field providing an economic and strategic view based on engineering design for mass customization in dynamic and uncertain market environments. For service sciences, the contributions of this method provide a new and visual representation for modeling and designing services. Especially, concepts from product family design methods yield insight into methods for modeling and designing a service family.

The SMPDM includes various decision-making methods that can be codified into an agent system, which enable more effective decision-making. The limitations of decision-making in the SMPDM are summarized as follows.

• The negotiation mechanism: To obtain agents’ knowledge, product analysis is performed through customer surveys and dissection for existing products. A method for identifying modules is necessary to perform task decomposition using the agents’ knowledge. Since a utility function is sensitive to parameters in the mathematical model, the parameters should be determined based on products’ characteristics, company’s and customers’ preferences, and the market environment. As increasing the number of modules in a product family, an effective search algorithm is needed to generate a set of feasible deals taking the agents’ negotiation.
• *The game theoretic approach*: To improve the proposed Bayesian game, the designers’ knowledge and customers’ requirements should be considered to establish module design strategies effectively. For large-scale product and service families, an effective search algorithm is needed to generate a set of feasible strategy in a game. To explore the best response of players in the Bayesian games, we needed to consider computationally intensive numerical integration such as Markov-chain or Monte-Carlo simulation methods for the market share ratios and customer’s preferences.

### 8.3 Recommendations and Future Work

This section summarizes recommendation from this research along with a discussion of the research limitations and future work.

**Knowledge Management for Product and Service Family**: The SMPDM gives a direction to manage information related to product and service family design for databases and design repositories in geographically distributed design environments. Knowledge-intensive and collaborative support has been increasingly important in product and service development to maintain and create future competitive advantages [100]. In knowledge support and management systems, data mining approaches facilitate extraction of information in design repositories to generate new knowledge for product and service development. Representing products and services within ontologies provides an approach to support data mining techniques by capturing, configuring, and reasoning
both linguistic and parametric design information in a knowledge management system effectively [58]. Regarding the SMPDM, future work should focus on discovering design rules to support a design knowledge system for product and service development, and expanding its application to agent-based design knowledge system in dynamic design environments.

**Module and Platform Identification for Configuration:** To facilitate family design, product and service configuration is important for increasing the variety of products and services offered. The proposed method for identifying modules and a platform can be applied to a product and service family design problems by modeling and defining modules based on configuration knowledge. Therefore, future research efforts should focus on considering functional requirements, reusability, and configurability in platform and module design, and extending its application to large databases or design repositories related to product and service design.

**Decision-Making for Strategic Platform Design:** Electronic markets have improved traditional product and service development processes by increasing the participation of customers and utilizing various recommender systems to help satisfy individual customer needs. Agent-based technologies provide a natural approach to integrate information related to design in a distributed environment. Decision-making methods in the SMPDM can be used to recommend a design solution by integrating distributed design information and knowledge. Future research efforts will focus on developing an agent-based design recommender system for customized recommendations in product and service development and enhancing agents’ knowledge to reflect various
market environments better. Future work should focus on improving the efficiency of the proposed Bayesian game, developing design strategies for various product and service family environments and expanding its application to develop a negotiation mechanism for web-based product and service family design.

**Extension to Software Family Design**: For mass customization, many products include functions or options that can be easily achieved via software to increase customer satisfaction. The SMPDM provides a foundation for applications related to identify modules and determine a platform design strategy in software family design as well. Future research efforts should focus on extending the SMPDM to software family design.

### 8.4 Concluding Remark

In product and service development for mass customization, design information and knowledge related to customers’ needs and functional requirements are continuous and dynamically changing within rapidly innovative and increasingly global market environments. Family design approaches have provided successful solutions for companies that should offer the variety of products and services by sharing and reusing assets such as components, information, processes, and knowledge across families of products and services. This dissertation provides applications and guidelines for future research work in this nascent field of engineering design through the SMPDM, which can help design new product and service families and advance the state-of-the-art in engineering design.
Bibliography


Appendix A

The Power Tools’ Components, Functional Model, FCM, and Functional Hierarchy

In this appendix, components, functional models, function component metrics (FCM), functional hierarchies for all the six power tools have been presented.

Table A-1: Components for a Power Tool Family

<table>
<thead>
<tr>
<th>Parts</th>
<th>Module Type</th>
<th>Jigsaw</th>
<th>Circular Saw</th>
<th>Brad Nailer</th>
<th>Drill</th>
<th>Sander</th>
<th>Flashlight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ball</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Base and Rotating Column</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Battery</td>
<td>Common</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Battery Terminal Connector</td>
<td>Common</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Blade Guide Device</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Blade Holder Washers (Tight turn)</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Blade Storage Cover</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Bulb</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Bulb Holder</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Chuck</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Connector</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Cover Cap</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Gear Assembly</td>
<td>Variant</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Gear Housing / Cover</td>
<td>Variant</td>
<td>1</td>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Grip</td>
<td>Variant</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Guide &amp; Mounting Pin</td>
<td>Unique</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Guiding Base Plate</td>
<td>Variant</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/L speed switch</td>
<td>Unique</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Magazine assembly</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Parts</td>
<td>Module Type</td>
<td>Jigsaw</td>
<td>Circular Saw</td>
<td>Brad Nailer</td>
<td>Drill</td>
<td>Sander</td>
<td>Flashlight</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------</td>
<td>--------</td>
<td>--------------</td>
<td>-------------</td>
<td>-------</td>
<td>--------</td>
<td>------------</td>
</tr>
<tr>
<td>Magazine Top cover Plates</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Motor</td>
<td>Variant</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Mounting Bracket</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mounting Brackets</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Nail Hitter plate</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Nozzle Cap</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Outer Casing (enclosure)</td>
<td>Variant</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>Plastic Head Screw</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Projector case</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Projector Lens</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Protection Cover</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Sander Base</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Saw Blade</td>
<td>Unique</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scotch yoke</td>
<td>Unique</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sliding Plate</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Solid Driver Block</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Spirit Level</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Spring</td>
<td>Variant</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spring Tension Adjuster</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Stopper for Solid Driver Block</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Suspension Plates</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Suspension Shafts</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Switch Assembly</td>
<td>Variant</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Switch connector</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trigger (Switch) Safety Mechanism</td>
<td>Variant</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vibration Generator</td>
<td>Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Washer</td>
<td>Variant</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>
Figure A-1: Functional Model of a Drill

Figure A-2: Functional Model of a Brad Nailer
Figure A-5: Functional Model of a Flash Light
### Table A-2: FCM for a Circular Saw

<table>
<thead>
<tr>
<th>Function/Component</th>
<th>Screws (outer enclosure)</th>
<th>Right Outer Case (enclosure)</th>
<th>Cover Cap</th>
<th>Screws</th>
<th>Grip</th>
<th>Plastic Head Screw</th>
<th>Spring</th>
<th>Bolt</th>
<th>Plastic Head Bolt</th>
<th>Umbrella Head Screws (long)</th>
<th>Umbrella Head Screws (small)</th>
<th>DC motor + Gear</th>
<th>Switch assembly (Continuous)</th>
<th>Battery Terminal Connector</th>
<th>Gear train assembly</th>
<th>Screws</th>
<th>Gear train housing Left</th>
<th>Screws</th>
<th>Screw (hex head) Left Outer Case, Base Plate, Blade Cover and Angular Scale</th>
<th>Blade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secure Solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Export Solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Regulate Solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Actuate Electrical Energy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Comfort Hand</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Convert Electrical Energy to</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mechanical Energy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distribute Mechanical Energy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distribute Heat Energy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Export Energy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Guide Solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Import Human Energy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Import Energy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Import hand</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Import Solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Regulate Electrical Energy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Separate Solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stop Slippage</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stop Solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Store Solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer Electrical energy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer Rot. Energy to Solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change Speed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------</td>
<td>-----------</td>
<td>------------------</td>
<td>----------------</td>
<td>---------------------</td>
<td>--------</td>
<td>--------------</td>
<td>---------------------</td>
<td>---------</td>
<td>---------------------</td>
<td>--------</td>
<td>----------------------</td>
<td>-------------------------------</td>
<td>-----------------------------</td>
<td>------</td>
<td>-----------</td>
<td>-------------------</td>
<td>--------------</td>
<td>---------------</td>
<td>--------</td>
</tr>
<tr>
<td>Actuate e.e.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change Solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Convert m.e.1 to m.e.2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distribute m.e.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Export Solid</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Guide Solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Guide Solid</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Import e.e.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Import Hand</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Import Solid</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Secure Solid</td>
<td>1 1 1 1 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 1 1 1 1 1 1 1 1 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sense Hand</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Separate Solid</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stop Solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Store Solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer e.e.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer e.e. to m.e.</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer h.e. to e.e.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer m.e.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table A-4: FCM for a Drill

<table>
<thead>
<tr>
<th>Function/Component</th>
<th>Screw</th>
<th>Right Case</th>
<th>Left Case</th>
<th>Spirit Level</th>
<th>DC Motor 76073 Johnson 3D2952</th>
<th>Continuous switch module + Safety cover</th>
<th>Battery terminal connector</th>
<th>Rotation direction switch</th>
<th>Washer</th>
<th>Gear 1</th>
<th>Gear 2</th>
<th>Gear Holder</th>
<th>Gear 3</th>
<th>Gear 4</th>
<th>Chuck</th>
<th>H/L speed switch</th>
</tr>
</thead>
<tbody>
<tr>
<td>secure solid</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indicate if the drill is levered</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Convert e.e. to m.e (rotational)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Regulate electricity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Actuate electricity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer electricity energy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change rotation direction</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Secure gears inside gear train</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change speed, change torque</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Position gear, transfer motion</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table A-5: FCM for a Sander

<table>
<thead>
<tr>
<th>Function\Component</th>
<th>Right Housing</th>
<th>Vibration Generator</th>
<th>DC Motor</th>
<th>Battery Terminal Connector</th>
<th>Power Switch Assembly</th>
<th>Wires</th>
<th>Mounting Brackets</th>
<th>Gripper</th>
<th>Swash Cover</th>
<th>Left Housing</th>
<th>Sander Base</th>
<th>Suspension Plates</th>
<th>Screws</th>
<th>Washers</th>
<th>Suspension Shafts</th>
<th>Screws</th>
<th>Screws</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actuate e.e.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Convert m.e.1 to m.e.2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distribute m.e.</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Export solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Import e.e.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Import hand</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Import solid</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Secure solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sense hand</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Separate solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stop solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer m.e.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer e.e.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer e.e. to m.e.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer h.e. to e.e</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table A-6: FCM for a Brad Nailer

| Function/Component                                      | Screws | Screws | Outer Right casing | Outer Left casing | Gap | Switch trigger + Washer + Spring | Switch connector | Spring | Spring Terminal Assembly | Washer | Spring (large) | Spring (large) | Motor + gear (16 teeth) | Gear Assembly | Solid Driver plate | Nail Housing plate | Switches | Magazine assembly | Sliding Plate | Battery + Terminal Connector | Magazine Top cover plates | Screw (hex) | Spring | Safety Attachment for switch | Spring | Nozzle Cap |
|--------------------------------------------------------|--------|--------|--------------------|------------------|-----|---------------------------------|------------------|--------|--------------------------|--------|-----------------|----------------|------------------------|----------------|------------------------|------------------|----------|------------------|--------------|--------------|
| Secure Solid                                           | 1      | 1      | 1                  | 1                | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Import hand                                            | 1      |        |                    |                  |      |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Import h.e                                             | 1      | 1      | 1                  |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Transport M.e to solid                                 | 1      | 1      | 1                  |                  |      |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Stop solid                                             |        |        | 1                  |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Regulate m.e                                           |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Activate electricity                                   |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Export M.e to solid                                    |        |        |                    |                  | 1   | 1                               |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| convert e.e to m.e (rotational)                        |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Change speed                                           |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Regulate e.e                                           |        |        |                    |                  | 1   | 1                               |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Transfer h.e to solid                                  |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Import m.e (rotational energy)                         |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Import Solid                                           |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Couple solid to solid                                  |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Transfer e.e                                           |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Export solid                                           |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Secure solid                                           |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Secure cover plates to magazine                        |        |        |                    |                  |      |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Regulate solid                                         |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Transfer energy                                        |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Guide solid                                            |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Comfort hand                                           |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
| Prevent Slipage                                        |        |        |                    |                  | 1   |                                 |                  |        |                          |        |                 |                |                        |                |                        |                  |          |                  |              |              |
## Table A-7: FCM for a Flash Light

<table>
<thead>
<tr>
<th>Component</th>
<th>Proj case</th>
<th>Proj lens</th>
<th>Bulb holder</th>
<th>Screw</th>
<th>Spring</th>
<th>Bulb</th>
<th>On/off Switch</th>
<th>Switch cover</th>
<th>Battery terminal</th>
<th>Connector</th>
<th>Wires</th>
<th>Outer casing</th>
<th>Screw</th>
<th>Spring</th>
<th>Ball</th>
<th>Connector</th>
<th>Screw</th>
<th>Screw</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import H. E.</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change angle</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Secure solid</td>
<td>1 1 1 1 1 1</td>
<td>1</td>
<td>1 1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer E. E</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Import Electricity</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Export light</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Import Hand</td>
<td>1 1 1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer H. E.into E. E</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer E. E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Position Solid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>
Figure A-6: Functional Hierarchy of a Circular Saw

Figure A-7: Functional Hierarchy of a Drill

Figure A-8: Functional Hierarchy of a Jig Saw
Figure A-9: Functional Hierarchy of a Brad Nailer

Figure A-10: Functional Hierarchy of a Sander

Figure A-11: Functional Hierarchy of a Flash Light
<table>
<thead>
<tr>
<th>Product</th>
<th>Customer Needs</th>
<th>Module</th>
<th>Function</th>
<th>Functional Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circular saw</td>
<td>Circle cutting - Torque: 12 N.m.</td>
<td>Electronics</td>
<td>import</td>
<td>Battery 5.0 A</td>
</tr>
<tr>
<td></td>
<td>Electric method - Lightweight and portable</td>
<td></td>
<td>activate</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Safety</td>
<td>Motor</td>
<td>convert</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Varying speed</td>
<td>Battery</td>
<td>transfer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Variable cut angle</td>
<td>Blade</td>
<td>import</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Input</td>
<td>import</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Gear train</td>
<td>import</td>
<td></td>
</tr>
<tr>
<td>Drill</td>
<td>Drilling - Torque: 15 N.m.</td>
<td>Electronic</td>
<td>import</td>
<td>Battery 7.0 A</td>
</tr>
<tr>
<td></td>
<td>Electric method</td>
<td>Motor</td>
<td>convert</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lightweight and portable</td>
<td>Battery</td>
<td>report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Safety</td>
<td>Gear train</td>
<td>transfer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Varying speed</td>
<td>Bit</td>
<td>import</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Quick blade change mechanism</td>
<td>Input</td>
<td>import</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Variable blade size</td>
<td>Gear train</td>
<td>transfer</td>
<td></td>
</tr>
<tr>
<td>Jig saw</td>
<td>Cutting - Torque: 7 N.m.</td>
<td>Electronic</td>
<td>import</td>
<td>Battery 5.0 A</td>
</tr>
<tr>
<td></td>
<td>Electric method</td>
<td>Motor</td>
<td>convert</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lightweight and portable</td>
<td>Motor</td>
<td>translate</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Safety</td>
<td>Conversion</td>
<td>translate</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Varying speed</td>
<td>Battery</td>
<td>report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Quick blade change mechanism</td>
<td>Gear train</td>
<td>transfer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Variable blade size</td>
<td>Input</td>
<td>import</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Blade</td>
<td>import</td>
<td></td>
</tr>
<tr>
<td>Nailer</td>
<td>Nailing</td>
<td>Electronic</td>
<td>import</td>
<td>Battery 7.0 A</td>
</tr>
<tr>
<td></td>
<td>Electric method</td>
<td>Motor</td>
<td>convert</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lightweight and portable</td>
<td>Motor</td>
<td>translate</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Safety</td>
<td>Nail hitter</td>
<td>convert</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Easy loading</td>
<td>Battery</td>
<td>import</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Variable size nails</td>
<td>Magazine</td>
<td>import</td>
<td></td>
</tr>
<tr>
<td></td>
<td>High capacity (less load)</td>
<td>Input</td>
<td>import</td>
<td></td>
</tr>
<tr>
<td>Sander</td>
<td>Sanding and smoothing</td>
<td>Electronic</td>
<td>import</td>
<td>Battery 5.0 A</td>
</tr>
<tr>
<td></td>
<td>Electric method</td>
<td>Motor</td>
<td>convert</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lightweight and portable</td>
<td>Motor</td>
<td>translate</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Safety</td>
<td>Conversion</td>
<td>convert</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ease in switching ON/OFF</td>
<td>Battery</td>
<td>translate</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Should not heat up fast</td>
<td>Input</td>
<td>import</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Variable speed</td>
<td>Magazine</td>
<td>import</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Safety</td>
<td>Input</td>
<td>import</td>
<td></td>
</tr>
</tbody>
</table>
Figure B-1: Functional Properties for a Circular Saw
Figure B-2: Assembly Properties a Circular Saw
Figure B-3: Product Specifics Level Properties a Circular Saw
### Table B-1: BOM of a Circular Saw

<table>
<thead>
<tr>
<th>Part Name</th>
<th>Function (Sub-fct. description from fct. model)</th>
<th>Mass (gm)</th>
<th>Finish</th>
<th>Color</th>
<th>Material</th>
<th>Manufacturing Process</th>
<th>Dimensions (in.)</th>
<th>DFM Cost Analysis Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Screws (outer enclosure)</td>
<td>Secure casings together</td>
<td>1.6</td>
<td>Powder Coated Black Steel Rolled</td>
<td>Dia 0.25&quot; Length 0.9&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Right Outer Case (enclosure)</td>
<td>Secure Parts, Import hand, Import Grip</td>
<td>260.4</td>
<td>None Dark Blue Plastic Injection Molding</td>
<td>7&quot; X 10&quot; X 2&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cover Cap</td>
<td>Store motor and Distribute Heat through vents</td>
<td>63.3</td>
<td>None Black Plastic Injection Molding</td>
<td>3.5&quot; X 2.75&quot; X 1.5&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Screws</td>
<td>Secure cap to the outer casing</td>
<td>1.8</td>
<td>Powder Coated Black Steel Rolled</td>
<td>Dia 0.25&quot; Length 1.06&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Grip</td>
<td>Import hand, comfort hand, prevent slipping, import h.e, distribute M.E., transfer h.e</td>
<td>30.7</td>
<td>None Black outer + White inner Rubber + Plastic Rubber Cutting + Injection Molding</td>
<td>4&quot; X 1.25&quot; X 1.5&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Plastic Head Screw</td>
<td>Secure Scale in place</td>
<td>5.4</td>
<td>None Black Head + Silver Steel Injection molded + Rolled</td>
<td>.875&quot; X 0.25&quot; X 0.875&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spring</td>
<td>Stop Solid</td>
<td>0.3</td>
<td>None Silver Steel wire drawing</td>
<td>Dia 0.375&quot; X length 0.5&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Umbrella Head Screw (long)</td>
<td>Secure Base Plate to the Outer Case</td>
<td>22.3</td>
<td>Powder Coated Black Steel Rolling + powder coating</td>
<td>Head Dia 0.375&quot; X length 4&quot; X screw dia 0.25&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bolt</td>
<td>Secure Base Plate to the Outer Case</td>
<td>6.8</td>
<td>Powder Coated Black Steel metal inner dia 0.25&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Plastic Head Bolt</td>
<td>secure blade for cutting at angle</td>
<td>6.8</td>
<td>None Black outer + Golden inner Plastic + Brass Injection molding + Tapping</td>
<td>1.25&quot; X 0.5&quot; X 0.5&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Umbrella Head Screw (small)</td>
<td>secure blade for cutting at angle</td>
<td>5.1</td>
<td>Powder Coated Black Steel Rolled</td>
<td>Dia 0.25&quot; X length 0.5&quot; X head dia 0.625&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DC motor + Gear RS775WC8514 TD293801 CCWIS</td>
<td>Convert e.e to m.e (rotational)</td>
<td>750</td>
<td>None Silver + Grey Aluminium and steel and various</td>
<td>4&quot; X 1.625&quot;X 1.625&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Switch assembly (Continuous Control)</td>
<td>Regulate electricity, actuate electricity</td>
<td>20</td>
<td>None Black Plastic + Various Various</td>
<td>2.125&quot; X 1.5&quot; X 0.625&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Battery Terminal Connector</td>
<td>Transfer Electrical energy</td>
<td>31.1</td>
<td>None Black + copper Plastic + Copper Injection molding + stamping</td>
<td>0.625&quot; X 1&quot; X 1.125&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gear train housing and assembly</td>
<td>Transfer rot. E to blade, import speed, export speed, Secure Gear</td>
<td>260.9</td>
<td>None Silver + brass Aluminium and steel Casting + milling</td>
<td>2.875&quot; X 2.25&quot;X 2.75&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Screws</td>
<td>Secure gear train to motor</td>
<td>1.2</td>
<td>Powder Coated Black Steel Rolled</td>
<td>0.75&quot; X 0.25&quot; X 0.15&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gear train housing Left</td>
<td>Secure gear assembly, Protect gears</td>
<td>100</td>
<td>None Silver Aluminium Casting</td>
<td>2&quot;X2&quot;X0.35&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blade Holder Washers (Tight turn)</td>
<td>Secure blade in position</td>
<td>35.2</td>
<td>None Brass Brass Turning</td>
<td>outer dia 1.125&quot; X inner dia 0.625&quot; X 0.375 thick</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Screw (Hex head)</td>
<td>Secure washer to gear train</td>
<td>6.3</td>
<td>Powder Coated Black Steel Rolled</td>
<td>head dia 0.625&quot; X screw dia 0.25&quot; X length 0.75&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Left Outer Case (enclosure) and Base Plate and Blade Cover and Angular Scale</td>
<td>Secure Parts, Import hand, Import Grip, Protect blade, prevent splinters from flying, Regulate cutting angle, guide movement on surface</td>
<td>2750</td>
<td>None Dark Blue Plastic Injection Molding</td>
<td>12&quot; X 10&quot; X 5&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table B-2: BOM of a Sander

<table>
<thead>
<tr>
<th>Part Name</th>
<th>Function (Sub-fct. description from fct. model)</th>
<th>Mass (gm)</th>
<th>Finish</th>
<th>Color</th>
<th>Material</th>
<th>Manufacturing Process</th>
<th>Dimensions (in.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Right Housing</td>
<td>Import Hand, Distribute Mechanical Energy, Secure Solid</td>
<td>198.7</td>
<td>None</td>
<td>Dark Blue</td>
<td>Plastic</td>
<td>Injection Moulding and Vulcanising</td>
<td>7.5&quot; X 5&quot; X 2&quot;</td>
</tr>
<tr>
<td>DC Motor</td>
<td>Transfer Electric Energy to Mechanical Energy</td>
<td></td>
<td>None</td>
<td>Silver</td>
<td>Aluminium, Copper etc.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Battery Terminal Connector</td>
<td>Transfer Electric Energy</td>
<td></td>
<td>None</td>
<td>Black, Copper</td>
<td>Plastic and Copper</td>
<td>Injection Moulding + Stamping</td>
<td>0.625&quot; X 1&quot; X 1.125&quot;</td>
</tr>
<tr>
<td>Power Switch Assembly</td>
<td>Actuate Electricity, Transfer Human Energy to Electric Energy, Import Electric Energy</td>
<td>189</td>
<td>None</td>
<td>Black, Silver</td>
<td>Plastic and Aluminium</td>
<td>Injecting Moulding, Pressing</td>
<td>0.5&quot; X 0.75&quot; X 0.5&quot;</td>
</tr>
<tr>
<td>Wires</td>
<td>Transfer Electric Energy</td>
<td></td>
<td>None</td>
<td>White, Blue, Red, Black, Yellow</td>
<td>Rubber and Copper Drawing</td>
<td></td>
<td>0.25&quot; Diameter</td>
</tr>
<tr>
<td>Mounting Brackets</td>
<td>Separate Solid, Secure Solid</td>
<td>5.6</td>
<td>None</td>
<td>Black</td>
<td>Plastic</td>
<td>Injection Moulding</td>
<td>3.5&quot; X 1.5&quot;</td>
</tr>
<tr>
<td>Gripper</td>
<td>Import Hand, Sense Hand, Secure Solid</td>
<td>23.3</td>
<td>None</td>
<td>Black</td>
<td>Plastic</td>
<td>Injection Moulding and Vulcanising</td>
<td>3.75&quot; X 2.5&quot;</td>
</tr>
<tr>
<td>Switch Cover</td>
<td>Stop Solid</td>
<td>0.9</td>
<td>None</td>
<td>Transparent</td>
<td>Rubber</td>
<td>Injection Moulding</td>
<td>0.8&quot; X 0.6&quot;</td>
</tr>
<tr>
<td>Left Housing</td>
<td>Import Hand, Distribute Mechanical Energy</td>
<td>192.7</td>
<td>None</td>
<td>Dark Blue</td>
<td>Plastic</td>
<td>Injection Moulding and Vulcanising</td>
<td>7.5&quot; X 5&quot; X 2&quot;</td>
</tr>
<tr>
<td>Sander Base</td>
<td>Import Solid, Secure Solid, Separate Solid, Export Solid</td>
<td>98.8</td>
<td>1/3&quot; Sand Paper</td>
<td>Black</td>
<td>Rubber and Plastic</td>
<td>Injection Moulding and Vulcanising</td>
<td>7.6&quot; X 3.6&quot; X 0.5&quot;</td>
</tr>
<tr>
<td>Suspension Plates</td>
<td>Separate Solid, Stop Solid</td>
<td>6.4</td>
<td>Coated</td>
<td>Black</td>
<td>Steel</td>
<td>Stamping</td>
<td>1.5&quot; X 1.75&quot;</td>
</tr>
<tr>
<td>Screws</td>
<td>Secure Solid</td>
<td>0.9</td>
<td>Coated</td>
<td>Black</td>
<td>Steel</td>
<td>Rolling</td>
<td>0.44&quot;X0.25&quot;X0.125&quot;</td>
</tr>
<tr>
<td>Washers</td>
<td>Stop Solid</td>
<td>0.6</td>
<td>Coated</td>
<td>Black</td>
<td>Steel</td>
<td>Stamping</td>
<td>0.25&quot;(Dia. Of Hole)</td>
</tr>
<tr>
<td>Suspension Shafts</td>
<td>Separate Solid, Stop Solid</td>
<td>6.2</td>
<td>Coated</td>
<td>Silver</td>
<td>Steel</td>
<td>Drawing, Bending</td>
<td>3&quot; X 0.1&quot; - 1&quot;(L-Angle)</td>
</tr>
<tr>
<td>Screws</td>
<td>Secure Solid</td>
<td>0.9</td>
<td>Coated</td>
<td>Black</td>
<td>Steel</td>
<td>Rolling</td>
<td>0.565&quot; x 0.25&quot;X0.125&quot;</td>
</tr>
<tr>
<td>Screws</td>
<td>Secure Solid</td>
<td>1.3</td>
<td>Coated</td>
<td>Black</td>
<td>Steel</td>
<td>Rolling</td>
<td>0.75&quot; X 0.25&quot; X 0.15&quot;</td>
</tr>
<tr>
<td>Part Name</td>
<td>Function (Sub-fct. description from fct. model)</td>
<td>Mass (gm)</td>
<td>Finish</td>
<td>Color</td>
<td>Material</td>
<td>Manufacturing Process</td>
<td>Dimensions (in.)</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------------</td>
<td>-----------</td>
<td>--------</td>
<td>-------</td>
<td>----------</td>
<td>----------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Scotch yoke</td>
<td>Convert m.e.1 to m.e.2</td>
<td>26.2</td>
<td>Coating</td>
<td>Black</td>
<td>Steel</td>
<td>Casting</td>
<td>4.375&quot; X 1.25&quot; X 0.25&quot;</td>
</tr>
<tr>
<td>Saw Blade</td>
<td>Import Solid, Export Solid, Separate Solid</td>
<td>5.5</td>
<td>Coating</td>
<td>Black</td>
<td>Steel</td>
<td>Stamping</td>
<td>4&quot; X 5/16&quot; X 1/32&quot;</td>
</tr>
<tr>
<td>Mounting Bracket</td>
<td>Secure Solid</td>
<td>10.8</td>
<td>Coating</td>
<td>Silver</td>
<td>Aluminium</td>
<td>Metal Casting</td>
<td>7/8&quot; X 1/2&quot; X 1/2&quot;</td>
</tr>
<tr>
<td>Hexagonal Bolt</td>
<td>Secure Solid</td>
<td>1.7</td>
<td>Coating</td>
<td>Black</td>
<td>Steel</td>
<td>Metal Casting</td>
<td>5/8&quot; X 1/4&quot;</td>
</tr>
<tr>
<td>Guide &amp; Mounting Pin</td>
<td>Secure Solid, Guide Solid</td>
<td>3.5</td>
<td>Coating</td>
<td>Black</td>
<td>Steel</td>
<td>Drawing</td>
<td>0.875&quot; X 0.19&quot;</td>
</tr>
<tr>
<td>Washers</td>
<td>Secure Solid</td>
<td>0.6</td>
<td>Coating</td>
<td>Black</td>
<td>Steel</td>
<td>Stamping</td>
<td>0.5&quot; Dia.</td>
</tr>
<tr>
<td>Left Housing</td>
<td>Import Hand, Distribute m.e.</td>
<td>202</td>
<td>Dark Blue</td>
<td>Plastic</td>
<td>Injection Moulding</td>
<td>7.5&quot; X 7&quot; X 1.125&quot;</td>
<td></td>
</tr>
<tr>
<td>Gear Shaft Assembly</td>
<td>Transfer m.e.</td>
<td>148.2</td>
<td>Coating</td>
<td>Silver</td>
<td>Steel</td>
<td>Milling</td>
<td>~80 teeth</td>
</tr>
<tr>
<td>DC Motor</td>
<td>Transfer e.e. to m.e.</td>
<td>~260</td>
<td>None</td>
<td>Silver</td>
<td>Aluminium, Copper etc.</td>
<td>Injection Moulding and Insertion</td>
<td>2.25&quot; X 2.5&quot; X 2.25&quot;</td>
</tr>
<tr>
<td>Motor Mounting Cover</td>
<td>Secure Solid</td>
<td>1.3</td>
<td>Coating</td>
<td>Black, Silver</td>
<td>Plastic and Steel</td>
<td>Injection Moulding and Insertion</td>
<td>0.875&quot; X 0.3&quot; X 0.15&quot;</td>
</tr>
<tr>
<td>Washers</td>
<td>Secure Solid</td>
<td>0</td>
<td>Coating</td>
<td>Black</td>
<td>Steel</td>
<td>Stamping</td>
<td>0.35&quot; X 0.25&quot;</td>
</tr>
<tr>
<td>ON/OFF Safety Device</td>
<td>Stop Solid</td>
<td>5.7</td>
<td>None</td>
<td>Black</td>
<td>Plastic</td>
<td>Injection Moulding</td>
<td>1.875&quot; X 1.25&quot; X 0.875&quot;</td>
</tr>
<tr>
<td>Continuous Power Switch Assembly</td>
<td>Actuate Electricity, Transfer h.e. to e.e., Import e.e.</td>
<td>101.9</td>
<td>None</td>
<td>Black, Silver</td>
<td>Plastic and Aluminium</td>
<td>Injection Moulding, Pressing</td>
<td>2.75&quot; X 1.5&quot; X 1&quot;</td>
</tr>
<tr>
<td>Battery Terminal Connector</td>
<td>Transfer e.e.</td>
<td>101.9</td>
<td>None</td>
<td>Black, Copper</td>
<td>Plastic and Copper</td>
<td>Injection Moulding + Stamping</td>
<td>0.625&quot; X 1&quot; X 1.125&quot;</td>
</tr>
<tr>
<td>Wires</td>
<td>Transfer e.e.</td>
<td>None</td>
<td>White, Blue, Red, Black, Yellow</td>
<td>Rubber and Copper</td>
<td>Drawing</td>
<td>0.25&quot; Diameter</td>
<td></td>
</tr>
<tr>
<td>Grip Cover</td>
<td>Import Hand, Sense Hand, Secure Solid</td>
<td>31.3</td>
<td>None</td>
<td>Black Rubber and white Solid</td>
<td>Plastic and Rubber</td>
<td>Injection Moulding and attaching Rubber</td>
<td>3.75&quot; X 1.25&quot; X 1&quot;</td>
</tr>
<tr>
<td>Blade Storage Cover</td>
<td>Store Solid</td>
<td>3.7</td>
<td>None</td>
<td>Dark Blue</td>
<td>Plastic</td>
<td>Injection Moulding</td>
<td>1.75&quot; X 1&quot; X 0.25&quot;</td>
</tr>
<tr>
<td>Right Housing</td>
<td>Import Hand, Distribute m.e., Secure Solid</td>
<td>204.8</td>
<td>None</td>
<td>Dark Blue</td>
<td>Plastic</td>
<td>Injection Moulding</td>
<td>7.5&quot; X 7&quot; X 1.125&quot;</td>
</tr>
<tr>
<td>Square Washer</td>
<td>Secure Solid</td>
<td>6.6</td>
<td>None</td>
<td>Black</td>
<td>Aluminium</td>
<td>Rolling</td>
<td>0.75&quot; X 0.5&quot; X 0.056&quot;</td>
</tr>
<tr>
<td>Screws</td>
<td>Secure Solid</td>
<td>1.6</td>
<td>Coating</td>
<td>Black</td>
<td>Steel</td>
<td>Rolling</td>
<td>1&quot; X 0.25&quot; X 0.125&quot;</td>
</tr>
<tr>
<td>Base and Rotating Column</td>
<td>Secure Solid, Change Solid</td>
<td>249.8</td>
<td>Coating</td>
<td>Silver</td>
<td>Steel</td>
<td>Casting + Spot Welding</td>
<td>6.5&quot;X3.8&quot; X 1/16&quot;(Base) 1.125&quot; Diameter</td>
</tr>
<tr>
<td>Blade Guide Device</td>
<td>Secure Solid, Guide Solid</td>
<td>28.1</td>
<td>Coating</td>
<td>Silver, Black Roller</td>
<td>Steel and Plastic</td>
<td>Casting + Riveting</td>
<td>3.25&quot; X 0.75&quot; X 0.375&quot;(Base) 0.5&quot; Diameter</td>
</tr>
<tr>
<td>Bolt</td>
<td>Secure Solid</td>
<td>1.9</td>
<td>Powder Coating</td>
<td>Black</td>
<td>Steel</td>
<td>Metal Casting</td>
<td>0.75&quot; X 0.56&quot; X 0.25&quot;</td>
</tr>
<tr>
<td>Washer</td>
<td>Secure Solid</td>
<td>1.3</td>
<td>Powder Coating</td>
<td>Silver</td>
<td>Steel</td>
<td>Stamping</td>
<td>1/2&quot; X 1/4&quot; X 1/16&quot;</td>
</tr>
<tr>
<td>Protection Cover</td>
<td>Stop Solid</td>
<td>12.5</td>
<td>None</td>
<td>Transparent Dark Brown</td>
<td>Plastic</td>
<td>Injection Moulding</td>
<td>2.75&quot; X 2.25&quot; X 1.25&quot;</td>
</tr>
</tbody>
</table>
### Table B-4: BOM of a Drill

<table>
<thead>
<tr>
<th>Part Name</th>
<th>Function (Sub-fct. description from fct. model)</th>
<th>Mass (gmi)</th>
<th>Finish</th>
<th>Color</th>
<th>DFM Cost Analysis Data</th>
<th>Dimensions (in.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Screw</td>
<td>Couple simetric covers</td>
<td>16</td>
<td>Powder coated</td>
<td>Black</td>
<td>Steel</td>
<td>Rolled</td>
</tr>
<tr>
<td>Spirit Level</td>
<td>Indicate if the drill is levered</td>
<td>20</td>
<td>Transparent Green</td>
<td>Plastic, liquid</td>
<td>Injection Molding</td>
<td>dia 3/8&quot; length 1 1/8&quot;</td>
</tr>
<tr>
<td>DC Motor 70073 Johnson 3D2952</td>
<td>Convert e.e. to m.e (rotational)</td>
<td>1000</td>
<td>None</td>
<td>Silver</td>
<td>Aluminum and Steel</td>
<td>Various</td>
</tr>
<tr>
<td>Continuous switch module + Safety cover</td>
<td>Regulate electricity, actuate electricity</td>
<td>70.5</td>
<td>Black</td>
<td>Plastic</td>
<td>Plastic</td>
<td>Various</td>
</tr>
<tr>
<td>Battery terminal connector</td>
<td>Transfer electricity energy</td>
<td>31.1</td>
<td>None</td>
<td>Black + Copper</td>
<td>Plastic + Copper</td>
<td>Injection molding + stamping</td>
</tr>
<tr>
<td>Rotation direction switch</td>
<td>Change rotation direction</td>
<td>4.1</td>
<td>None</td>
<td>Black</td>
<td>Plastic</td>
<td>Injection molding</td>
</tr>
<tr>
<td>Washer</td>
<td>Secure gears inside gear train</td>
<td>5.5</td>
<td>None</td>
<td>Black</td>
<td>Iron</td>
<td>Stamping</td>
</tr>
<tr>
<td>Gear 1</td>
<td>Change speed, change torque</td>
<td>37.3</td>
<td>None</td>
<td>Black</td>
<td>Steel</td>
<td>Milling</td>
</tr>
<tr>
<td>Gear 2</td>
<td>Change speed, change torque</td>
<td>6</td>
<td>None</td>
<td>Black</td>
<td>Steel</td>
<td>Milling</td>
</tr>
<tr>
<td>Gear Holder</td>
<td>Position gear, transfer motion</td>
<td>17.3</td>
<td>None</td>
<td>Black</td>
<td>Steal</td>
<td>Milling</td>
</tr>
<tr>
<td>Gear 3</td>
<td>Change speed, change torque</td>
<td>4</td>
<td>None</td>
<td>Black</td>
<td>Steel</td>
<td>Milling</td>
</tr>
<tr>
<td>Gear 4</td>
<td>Change speed, change torque</td>
<td>5.7</td>
<td>None</td>
<td>White</td>
<td>Plastic</td>
<td>Injection Molding</td>
</tr>
<tr>
<td>Chuck</td>
<td>Change speed, position tool</td>
<td>500</td>
<td>None</td>
<td>Black</td>
<td>Plastic and metal</td>
<td>Various</td>
</tr>
<tr>
<td>H/L speed switch</td>
<td>Change rotation speed</td>
<td>4.8</td>
<td>None</td>
<td>Black</td>
<td>Plastic</td>
<td>Injection Molding</td>
</tr>
<tr>
<td>Motor Cover</td>
<td>Secure motor</td>
<td>31.5</td>
<td>None</td>
<td>Black</td>
<td>Injection Molding</td>
<td>dia 1 3/4&quot; X length 3/8&quot;</td>
</tr>
</tbody>
</table>
Table B-5: BOM of a Brad Nailer

<table>
<thead>
<tr>
<th>Part Name</th>
<th>Function (Sub-kt. description from Kt. model)</th>
<th>Mass (gm)</th>
<th>Finish</th>
<th>Color</th>
<th>DFM Cost Analysis Data</th>
<th>Material</th>
<th>Manufacturing Process</th>
<th>Dimensions (in.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Screws</td>
<td>Secure outer right casing</td>
<td>1.3</td>
<td>Powder Coated</td>
<td>Black Steel</td>
<td>Rolled</td>
<td>dia 0.125&quot; X length 0.75&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Screw</td>
<td>Secure outer right casing</td>
<td>0.48</td>
<td>Powder Coated</td>
<td>Black Steel</td>
<td>Rolled</td>
<td>dia 0.1&quot; X length 0.5&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Outer Right Casing</td>
<td>Secure inner parts</td>
<td>238.3</td>
<td>None</td>
<td>Dark Blue Plastic</td>
<td>Injection molded</td>
<td>1.5&quot; X 1.125&quot; X 2&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Screw (Left side)</td>
<td>Secure outer left casing</td>
<td>0.4</td>
<td>Powder Coated</td>
<td>Black Steel</td>
<td>Rolled</td>
<td>dia 0.1&quot; X length 0.5&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Outer Left Casing</td>
<td>Secure inner parts</td>
<td>222</td>
<td>None</td>
<td>Dark Blue Plastic</td>
<td>Injection molded</td>
<td>1.5&quot; X 1.125&quot; X 2&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Grip</td>
<td>Import hand, Comfort hand, Prevent Slippage, import h.e, distribute M.e, Transfer h.e</td>
<td>30.9</td>
<td>None</td>
<td>Black outer + white inner</td>
<td>Rubber + plastic Rubber Cutting + injection molding</td>
<td>4&quot; X 1.25&quot; X 1.5&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Switch trigger + Washer + Spring</td>
<td>Import h.e, Actuate electricity, Regulate electricity</td>
<td>8.2</td>
<td>None</td>
<td>Black + Silver Plastic + Aluminium</td>
<td>Injection molding + wire drawing</td>
<td>2.25&quot;X 2.125&quot;X 0.5&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Switch connector</td>
<td>Transport M.e to toggle switches simultaneously</td>
<td>1.3</td>
<td>None</td>
<td>white Plastic + Aluminium</td>
<td>Injection molding</td>
<td>2.625&quot;X 0.625&quot;X 0.375&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spring</td>
<td>Stop sold</td>
<td>0.6</td>
<td>None</td>
<td>Silver Steel</td>
<td>Wire drawing</td>
<td>dia 0.1&quot; X length 1.125&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spring Tension Adjuster</td>
<td>Regulate m.e (tension)</td>
<td>23.7</td>
<td>None</td>
<td>Black Plastic + Aluminium</td>
<td>Injection molding</td>
<td>2.75&quot;X 1.25&quot;X 1.25&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Washer</td>
<td>Secure spring tension adjuster</td>
<td>1.5</td>
<td>None</td>
<td>Black Steel</td>
<td>Press working and Stamping</td>
<td>dia 0.75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spring (bale)</td>
<td>Export M.e (translational) to solid driver block, Import rotational Energy</td>
<td>35</td>
<td>None</td>
<td>Black Steel</td>
<td>Wire Drawing</td>
<td>dia 0.625&quot; X length 4.625&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Motor (70073 Johnson 3D3031) + gear (16 teeth)</td>
<td>convert e.e to m.e (rotational)</td>
<td>1000</td>
<td>None</td>
<td>Silver + grey Aluminium + Steel + various Various</td>
<td>Various</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gear Assembly</td>
<td>Charge speed</td>
<td>306</td>
<td>None</td>
<td>Black Plastic + steel</td>
<td>Injection molding + malling</td>
<td>2.5&quot; X 2&quot; X 2&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Solid Driver Block</td>
<td>Transfer m.e to nail hitter plate</td>
<td>186.9</td>
<td>None</td>
<td>Silver Grey Steel</td>
<td>Casting</td>
<td>2&quot; X 1.625&quot;X 0.875&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nail Hitter plate</td>
<td>Transfer m.e to the nail head</td>
<td>4.6</td>
<td>None</td>
<td>Silver Steel</td>
<td>Forming + Press Working</td>
<td>3.25&quot;X 0.03&quot;X 0.5&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Switch</td>
<td>Regulate e.e to motor</td>
<td>13.6</td>
<td>None</td>
<td>Black + magenta Plastic outer</td>
<td>Various</td>
<td>0.75&quot;X 0.25&quot;X 0.5&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Switch</td>
<td>Regulate e.e to motor (3 step)</td>
<td>30</td>
<td>None</td>
<td>Grey + Black Plastic + Styrofoam</td>
<td>Various</td>
<td>1.0625&quot; X 0.875&quot;X 0.5&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Magazine assembly</td>
<td>Import Nail, Import h.e, Export nail</td>
<td>188.7</td>
<td>None</td>
<td>Black + Transparent Plastic</td>
<td>Various</td>
<td>7.625&quot;X 6.75&quot;X 1.12&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sliding Plate</td>
<td>Couple nail hitter plate and solid driver block, Guide nail hitter plate and solid driver block</td>
<td>53.6</td>
<td>None</td>
<td>Black Steel</td>
<td>Bending + Stamping</td>
<td>3.125&quot;X 1.5&quot;X 0.375&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Battery Terminal Connector</td>
<td>Transfer e.e</td>
<td>31.1</td>
<td>None</td>
<td>Black + Copper Plastic + copper</td>
<td>Injection Molding + Stamping</td>
<td>0.625&quot;X 1&quot;X 1.125&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Magazine Top cover Plates</td>
<td>Stop m.e being pushed above nozzle, Position nail in nozzle</td>
<td>11.9/8.2</td>
<td>None</td>
<td>Black Steel</td>
<td>Stamping</td>
<td>(1.75&quot;X 0.875&quot;X 0.1&quot;) + (1.0625&quot;X 0.6875&quot;X 0.1&quot;)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Screw (hex)</td>
<td>Secure cover plates to magazine</td>
<td>1.4</td>
<td>None</td>
<td>Black Steel</td>
<td>Powder coated + rolled</td>
<td>Dia 0.125&quot; X length 0.375&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spring</td>
<td>Secure cover plates to magazine</td>
<td>0.2</td>
<td>None</td>
<td>Black Steel</td>
<td>Wire drawing</td>
<td>Dia 0.22&quot; X length 0.5025&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Safety Attachment for switch</td>
<td>Regulate switch</td>
<td>2.5</td>
<td>None</td>
<td>Black Plastic</td>
<td>Injection Molding</td>
<td>1.625&quot;X 0.625&quot;X 0.75&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spring</td>
<td>Transfer energy</td>
<td>0.1</td>
<td>None</td>
<td>Silver Steel</td>
<td>Wire drawing</td>
<td>dia 0.1875&quot;X Length 0.5&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nozzle Cap</td>
<td>Import solid, Guide nail</td>
<td>3</td>
<td>None</td>
<td>Transparent Plastic</td>
<td>Injection molding</td>
<td>0.75&quot;X0.5&quot;X 0.3125&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stopper for Solid Driver Block</td>
<td>Stop Solid block from hitting magazine side</td>
<td>7.1</td>
<td>None</td>
<td>Black Plastic</td>
<td>Injection molding</td>
<td>1.5625&quot;X 0.5&quot;X 0.3125&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part Name</td>
<td>Function (Sub-fct. description from fct. model)</td>
<td>Mass (g)</td>
<td>Finish</td>
<td>Color</td>
<td>DFM Cost Analysis Data</td>
<td>Material</td>
<td>Manufacturing Process</td>
<td>Dimensions (in.)</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------------------------</td>
<td>----------</td>
<td>---------</td>
<td>------------------------</td>
<td>------------------------</td>
<td>----------</td>
<td>-----------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Projector Lense</td>
<td>Secure solid (protects bulbs)</td>
<td>56.6</td>
<td>none</td>
<td>Transparent and Black</td>
<td></td>
<td>Plastic</td>
<td>Injection Molding</td>
<td>Dia 3.5&quot;</td>
</tr>
<tr>
<td>Bulb Holder</td>
<td>Secure solids (bulbs)</td>
<td></td>
<td></td>
<td>Black</td>
<td></td>
<td>Plastic</td>
<td>Injection Molding</td>
<td>Dia 1/18&quot;</td>
</tr>
<tr>
<td>On/Off Switch</td>
<td>Import Hand, Transfer H. E into E.E</td>
<td></td>
<td>none</td>
<td>Black</td>
<td></td>
<td>Steel + Plastic</td>
<td>Various</td>
<td>1&quot; x 1/4&quot; x 1/2&quot;</td>
</tr>
<tr>
<td>Battery terminal Connector</td>
<td>Transfer E. E</td>
<td></td>
<td>none</td>
<td>Black + Copper</td>
<td></td>
<td>Plastic + Copper</td>
<td>Injection molding and Stamping</td>
<td>0.625&quot; x 1&quot; x 1.125&quot;</td>
</tr>
<tr>
<td>Wires</td>
<td>Transfer E.E</td>
<td>71.2</td>
<td>none</td>
<td>White, blue, red, black, yellow</td>
<td>Rubber + Copper</td>
<td>Plastic</td>
<td>Injection Molding</td>
<td>Dia 1/4&quot;</td>
</tr>
<tr>
<td>Projector case</td>
<td>Import H. E., Change angle, secure solid (Bulbs, wires, plastic, lense, etc)</td>
<td></td>
<td>none</td>
<td>Black</td>
<td></td>
<td>Plastic</td>
<td>Injection Molding</td>
<td>Dia 4.75&quot;</td>
</tr>
<tr>
<td>Connector</td>
<td>Position solid (Projector case and Outer case)</td>
<td></td>
<td>none</td>
<td>Plastic</td>
<td></td>
<td>Plastic</td>
<td>Injection Molding</td>
<td>Dia 4/8&quot; X 1/8&quot;</td>
</tr>
<tr>
<td>Screw</td>
<td>Secure solid</td>
<td>0.5</td>
<td>none</td>
<td>Black</td>
<td></td>
<td>Steel</td>
<td>Rolled</td>
<td>Dia 1/8&quot; length 1/4&quot;</td>
</tr>
<tr>
<td>Spring</td>
<td>Secure Solid (bulb), transfer E.E</td>
<td>0.3</td>
<td>none</td>
<td>Silver</td>
<td></td>
<td>Iron</td>
<td>Wire drawing</td>
<td>Dia 0.1&quot; X length 1.125&quot;</td>
</tr>
<tr>
<td>Spring</td>
<td>Position Solid (Ball)</td>
<td>0.1</td>
<td>none</td>
<td>Silver</td>
<td></td>
<td>Iron</td>
<td>Wire drawing</td>
<td>Dia 0.22&quot; X length 0.75&quot;</td>
</tr>
<tr>
<td>Ball</td>
<td>Position Solid (Ball)</td>
<td>0.1</td>
<td>none</td>
<td>Silver</td>
<td></td>
<td>Silver</td>
<td>Various</td>
<td>dia 1/18&quot;</td>
</tr>
<tr>
<td>Screw</td>
<td>Secure Solid (Projector case and Outer Case)</td>
<td>3.0</td>
<td>none</td>
<td>Black</td>
<td></td>
<td>Steel</td>
<td>Rolled</td>
<td>dia 1/4&quot; X length 1/2&quot;</td>
</tr>
<tr>
<td>Bulb</td>
<td>Import Electricity, Export light</td>
<td>2.0</td>
<td>none</td>
<td>Transparent</td>
<td></td>
<td>Various</td>
<td>Various</td>
<td>18 v, 0.855 Amp</td>
</tr>
<tr>
<td>Switch cover</td>
<td>Import Hand, Secure Solid (switch)</td>
<td>2.8</td>
<td>none</td>
<td>Black</td>
<td></td>
<td>Plastic</td>
<td>Injection Molding</td>
<td>1 1/4&quot; x 3/4&quot; x 1/4&quot;</td>
</tr>
<tr>
<td>Left Outer Casing</td>
<td>Import Hand, Secure Solid (wires, battery connector and battery conector)</td>
<td>66.9</td>
<td>none</td>
<td>Blue</td>
<td></td>
<td>Plastic</td>
<td>Injection Molding</td>
<td>5&quot; X 2&quot; x 3/4&quot;</td>
</tr>
<tr>
<td>Right Outer Casing</td>
<td>Import Hand, Secure Solid (wires, battery connector and battery conector)</td>
<td>66.9</td>
<td>none</td>
<td>Blue</td>
<td></td>
<td>Plastic</td>
<td>Injection Molding</td>
<td>5&quot; X 2&quot; x 3/4&quot;</td>
</tr>
<tr>
<td>Screw</td>
<td>Secure Solid (outer Case)</td>
<td>3.0</td>
<td>none</td>
<td>Black</td>
<td></td>
<td>Steel</td>
<td>Rolled</td>
<td>dia 1/4&quot; X length 1/2&quot;</td>
</tr>
<tr>
<td>Product</td>
<td>Function</td>
<td>Medium</td>
<td>Attribute codes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>---------------</td>
<td>--------</td>
<td>-----------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Greater saw</td>
<td>Electronic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Motor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Block</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Gear train</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conversion</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Magazine</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jig saw</td>
<td>Electronic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Motor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Block</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Magazine</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nailer</td>
<td>Electronic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Motor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Magazine</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sander</td>
<td>Electronic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Motor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Magazine</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table B-7: Product Representation for a Power Tool Family**

<table>
<thead>
<tr>
<th>Product</th>
<th>Function</th>
<th>Medium</th>
<th>Attribute codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Greater saw</td>
<td>Electronic</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Motor</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Block</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Gear train</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conversion</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Magazine</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jig saw</td>
<td>Electronic</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Motor</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Block</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Magazine</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nailer</td>
<td>Electronic</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Motor</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Magazine</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sander</td>
<td>Electronic</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Motor</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Magazine</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table B-7: Product Representation for a Power Tool Family**

<table>
<thead>
<tr>
<th>Product</th>
<th>Function</th>
<th>Medium</th>
<th>Attribute codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Greater saw</td>
<td>Electronic</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Motor</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Block</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Gear train</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conversion</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Magazine</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jig saw</td>
<td>Electronic</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Motor</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Block</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Magazine</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nailer</td>
<td>Electronic</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Motor</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Magazine</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sander</td>
<td>Electronic</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Motor</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Magazine</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix C

Service Ontology for the Checking Account Services

Figure C-1: Process Modules of Deposit Function
Figure C-2: Service Components of Make a Deposit and Withdraw process modules
Figure C-3: Service Process Modules related to Request Service Components
VITA

Seung Ki Moon

Seung Ki Moon is a Graduate Research Assistant in the Harold and Inge Marcus Department of Industrial and Manufacturing Engineering at The Pennsylvania State University working Dr. Timothy W. Simpson and Dr. Soundar R. T. Kumara and in Engineering Design and Optimization Group (EDOG) and Laboratory of Intelligent Systems and Quality (LISQ). He received the B.S. and M.S. degrees in industrial engineering from Hanyang University, Ansan and Seoul, South Korea, in 1992 and 1995, respectively. Before Ph.D. student, he worked as Senior Research Engineer at the Hyundai Motor Company, Seoul, South Korea. His research interests focus on Product and Service Family and Platform Design, Intelligent Information Systems and Management, Strategic and Multidisciplinary Design Optimization, Data Mining, Knowledge Engineering in Products and Services. He also helped teach a undergraduate level class in the department of Mechanical and Nuclear Engineering.