Intelligent hybrid approach to product design conceptualisation

Alisantoso, Darius

2006

Alisantoso, D. (2006). Intelligent hybrid approach to product design conceptualisation.
Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/5666>

<https://doi.org/10.32657/10356/5666>

Nanyang Technological University

Downloaded on 26 Apr 2023 20:38:44 SGT

An Intelligent Hybrid Approach to Product Design Conceptualisation

Darius Alisantoso

School of Mechanical and Aerospace Engineering

A thesis submitted to the Nanyang Technological University
in fulfilment of the requirement for the degree of
Doctor of Philosophy

2006

TS
171
D218
2006

Acknowledgements

The author is indebted to Nanyang Technological University (NTU) for awarding the research scholarship for this work. The author would like to express his appreciation to all those that have rendered assistance and support to his Ph.D. work at NTU. First and foremost, the author would like to convey his sincerest gratitude and appreciation to his supervisor, Prof. Khoo Li Pheng, for his continuous and ceaseless guidance, support and constructive criticism. The author would also like to express his gratitude for his former supervisors, Dr Fok Sai Cheong and Dr Lee Boon Hong. Their insights have proven to be invaluable. The author would like to thank Dr Mathew C. Pritchard, Dr Stuart J. Passey and A/P Chen Chun Hsien for their practical feedback and suggestions. In addition, the author would also like to thank all the technicians in Design Research Center (DRC), in particular Mr. Chia Yak Khoong for his continual technical support. Finally, the author would like to express his heartfelt appreciation for his wife and parents for their understanding, encouragement and morale support.

Table of Contents

Acknowledgements	i
Table of Contents	ii
Abstract	viii
List of Publications	ix
List of Figures	x
List of Tables	xii
Chapter 1 Introduction	1
1.1. Background of the Research	1
1.1.1. Collaborative Design Trend	1
1.1.2. Collaborative Design in Conceptual Design	3
1.1.3. Design Concept-Capability Mapping	5
1.2. Objectives of the Research	6
1.3. Scope of the Research	7
1.4. Organisation of the Thesis	8
Chapter 2 Literature Review	10
2.1. Introduction	10
2.2. General Product Design	10
2.3. Collaborative Design System	12
2.3.1. Introduction to Collaborative Design	12
2.3.2. Collaborative Design System Functionalities	15
2.3.2.1. Tracking and Monitoring Decision Process	15
2.3.2.2. Anticipating Problems and Suggesting Solutions	16

2.3.2.3. Managing Data Sharing	18
2.3.2.4. Utilising External Information Sources	19
2.3.3. Virtual Reality Enhanced Design Environment.....	20
2.4. Design Chain.....	21
2.5. Conceptual Design Phase.....	23
2.5.1. Introduction to Conceptual Design Phase.....	23
2.5.2. Design Knowledge Representation, Design Rationale and Design Purpose	25
2.5.2.1. Functional Representation.....	27
2.5.2.2. Design Rationale Representation	29
2.6. Artificial Intelligence Approaches in Design.....	30
2.6.1. Intelligent Tools in Design.....	30
2.6.2. Artificial Intelligence	31
2.6.3. Knowledge Based System.....	34
2.6.4. Case Based Reasoning	35
2.6.5. Artificial Neural Network	36
2.7. Rough Set Theory	37
2.7.1. Introduction to Rough Set Theory	37
2.7.2. Incomplete Information.....	41
2.7.3. Non-Discrete Valued Attributes.....	43
2.7.4. Multiple Decision Attributes.....	45
2.7.5. Incremental Updating of Rules	46
2.8. Discussions.....	47
2.9. Summary	53
Chapter 3 Purpose Behaviour Structure Representation.....	57

3.1. Introduction	57
3.2. Functional Representation for Collaborative Product Development	58
3.3. Purpose Behaviour Structure Representation	61
3.3.1. Basic Notions	61
3.3.2. Structure Domain	62
3.3.3. Behaviour Domain	64
3.3.4. Purpose Domain	66
3.3.5. Discussion	68
3.4. A PBS Based Framework for Collaborative Product Development	72
3.5. A Case Study	75
3.6. Summary	79
Chapter 4 Design Concept-Capability Mapping	81
4.1. Introduction	81
4.2. Multiple Decision Attributes	83
4.2.1. Representation	83
4.2.2. Attributes Decomposition Algorithm for Multiple Decision Attributes	86
Attributes Decomposition Algorithm	86
4.2.4. A Proof of the Attributes Decomposition Algorithm	92
4.3. Incomplete Information	97
4.3.1. Background of Incomplete Information	97
4.3.2. Dissimilar Objects Algorithm for Incomplete Information	97
4.3.3. A Proof for the Dissimilar Objects Algorithm	100
4.3.4. Discussion on the Dissimilar Objects Algorithm	104
4.4. Non-Discrete Valued Attributes	106

4.4.1. MD-Heuristic Algorithm.....	106
4.4.2. Extended MD-Heuristic Algorithm	111
4.5. Summary	113
Chapter 5 Framework of Conceptual Design Assistant.....	115
5.1. Introduction.....	115
5.2. Design Analysis	116
5.2.1. Past Design Retriever.....	118
5.2.2. Design Capability Assessor	123
5.3. Design Synthesis.....	125
5.4. Incremental Updating of Rules	127
5.4.1. Approaches to Updating of Design Rule Base.....	127
5.4.1.1. Ad-hoc Updating Approach.....	127
5.4.1.2. Direct Updating Approach.....	129
5.4.1.3. The Incremental Updating Approach.....	131
5.4.1.4. Comparison of the Various Approaches to Updating of Rules.....	134
5.4.2. Details of the Incremental Updating Approach	135
5.4.2.1. The Algorithm.....	135
5.4.2.2. Illustration	136
5.4.2.3. Discussion on the Approach for Incremental Updating of Rules	142
5.5. A Framework for a Prototype Conceptual Design Assistant	143
5.5.1. Design Concept-Capability Mapping Mechanism.....	143
5.5.2. Past Design Retriever.....	144
5.5.3. The GUI	145
5.5.4. Design Capability Assessor	145

5.5.5. Design Concept Extractor	146
5.5.6. Incremental Updater.....	147
5.6. Operational Interactions of the Modules.....	148
5.7. Summary	150
Chapter 6 System Implementation and Case Studies.....	152
6.1. System Implementation.....	152
6.1.1. Overall System	152
6.1.2. Databases	154
6.1.3. Updating of Past Design	154
6.1.4. Updating of Global Discernibility Matrices.....	155
6.1.5. Design Analysis	156
6.1.6. Design Synthesis	157
6.2. Case Study 1: Elastomer Mold Design	159
6.2.1. Descriptions	159
6.2.2. Results and Discussions.....	162
6.3. Case Study 2: Cellular Phones Design.....	165
6.3.1. Descriptions	165
6.3.2. Results and Discussions.....	168
6.4. Summary	174
Chapter 7 Conclusions and Future Work.....	176
7.1. Conclusions.....	176
7.2. Future Work	181
References.....	183
Appendix 1: Case Study 1 Design Rules from Expert.....	200

Appendix 2: Case Study 1 Raw Data..... 201

Appendix 3: Case Study 2 Raw Data..... 204

Abstract

The increasing demand for better and cheaper products leads to the quest for even better designs as part of the overall product development strategy. Accordingly, companies in different geographical locations need to collaborate in order to realise shorter product life cycle, and hence shorter time to market. In order to assist designers in meeting this challenge in product development, conceptual design in particular, this work attempts to establish an intelligent hybrid approach to design analysis and design synthesis. The work proposes a new conceptual design paradigm known as design concept-capability mapping where past design data are used for the extraction of design rules. To facilitate the capturing of design knowledge comprising both functional design purpose and non-functional design purpose, a comprehensive design representation scheme known as Purpose-Behaviour-Structure has been proposed and developed. Past designs represented using the representation scheme are accessed by a rough set based approach for design rule extraction. The rough set based approach developed in this work is able to deal with four main issues in rule extraction, namely incomplete information, non-discrete valued attributes, multiple decision attributes and incremental updating of rules. The approach is supported by mathematical proofs. Based on the approach and representation scheme developed, a framework of the conceptual design assistant (CDA) is proposed and realised. Two case studies are used to illustrate the capability of the CDA. It has been established that the CDA is able to fulfil the functions of design analysis and design synthesis. With the incorporation of the ability to update design rules incrementally, the CDA is expected to assist designers in design concepts analysis and generation at the earliest stage of product development.

List of Publications

Alisantoso, D., Khoo, L.P., Lee, I.B.H., Fok, S.C. (2005) A rough set approach to design concept analysis in design chain, *International Journal of Advanced Manufacturing Technology*, 26, 427-435.

Alisantoso, D., Khoo, L.P., Lee, I.B.H. (2004) An approach to the analysis of design concepts using rough set theory, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 18, 343-355.

Alisantoso, D., Khoo, L.P., Lee, I.B.H., Lu, W.F. (2005) A design representation scheme for collaborative product development, *International Journal of Advanced Manufacturing Technology*, (in print).

Alisantoso, D. and Khoo, L.P. (2005) Product Conceptualisation: A Rough Set Based Data Mining Approach, (submitted to Journal of Data and Knowledge Engineering).

Alisantoso, D. Khoo, L.P., Lu, W.F. (2005) Early design analysis using rough set theory, *Intelligent Production Machines and Systems (IPROMS)*, (held in July 2005).

Alisantoso, D. Khoo, L.P., Rough Set Approach for Design Conceptualization with Incremental Updating Capability, (in preparation).

List of Figures

Figure 1.1. Transformation from Traditional to Collaborative Design Practice.....	2
Figure 1.2. Collaborative Design in Design Process	3
Figure 1.3. Design Concepts, Capabilities and Specifications Relationships (Alisantoso et al., 2004a).....	5
Figure 2.1. An electronic design chain (Martin and Schirrmeister, 2002).....	21
Figure 2.2. A Simplified Flow Chart for Conceptual Design	24
Figure 3.1. Functional Design Purpose and Non-Functional Design Purpose.....	60
Figure 3.2. A Typical PBS Representation.....	62
Figure 3.3. Different Ways to Represent a Design Purpose	70
Figure 3.4. A PBS based Framework for Collaborative Product Development	74
Figure 3.5. Realising Design Purposes through Different Modules	74
Figure 3.6. A FBS Representation for a Vacuum Cleaner.....	76
Figure 3.7. A PBS Representation for a Vacuum Cleaner.....	77
Figure 4.1. Design Concept-Capability Mapping Representations.....	82
Figure 5.1. CDA and Conceptual Design	115
Figure 5.2. Design Rule Generation.....	117
Figure 5.3. Design Analysis using the Design Rules Generated	117
Figure 5.4. Design Synthesis using Design Concept-Capability Mapping.....	126
Figure 5.5 Ad-hoc Updating Approach.....	128
Figure 5.6 Direct Updating Approach.....	129
Figure 5.7. Layers of A Global Discernibility Matrix	132
Figure 5.8 Approach for Incremental Updating of Rules	133

Figure 5.9. An Overall Framework for CDA.....	144
Figure 5.10. Design Capability Assessor and User Interactions.....	146
Figure 5.11. Design Concept Extractor and User Interactions	147
Figure 5.12. Interactions between Modules	149
Figure 6.1. System Overview: User Interfaces, Modules and Databases	152
Figure 6.2. Past Design Update Interface	155
Figure 6.3. Concept Input Interface	156
Figure 6.4. Capability Display Interface.....	157
Figure 6.5. Specification Input Interface	158
Figure 6.6. Concept Display Interface	159
Figure 6.7. PBS Representation of Elastomer Mold Design.....	160
Figure 6.8. PBS Representation for the Cellular Phones	167

List of Tables

Table 2.1. Advantages and Disadvantages of Different Knowledge Representation Schemes (Adapted from Turban, 1992).....	32
Table 2.2. Sample Information System with Incomplete Information	51
Table 3.1. Comparison of the Various Functional Representation Schemes.....	59
Table 3.2. Comparison between FBS and PBS.....	71
Table 4.1. An Information System with Multiple Decisions	84
Table 4.2. Relative Discernibility Matrix for Decision Attribute D	88
Table 4.3 Relative Discernibility Matrix for Decision Attribute E.....	88
Table 4.4. Rules Based on Decision Attribute D	89
Table 4.5. Rules Based on Decision Attribute E.....	90
Table 4.6. Complete Rules based on Decision Attributes D AND E.....	91
Table 4.7. Combinations of d_1 and d_2 using the Attributes Decomposition Algorithm..	95
Table 4.8. Possible Combinations of d_1 and d_2 for Rules Generated by Considering d_1 and d_2 Simultaneously.....	96
Table 4.9. An Information System with Incomplete Information.....	97
Table 4.10. The Information System after Replacing Incomplete Information with 'UA'	99
Table 4.11. Rules Derived from the Information System with Incomplete Information	99
Table 4.12. Final Rule Set after Elimination of 'UA'	99
Table 4.13. Information System after Eliminating 'UA'	104
Table 4.14. Rule set Derived from the Simplified Information System	105
Table 4.15. An Information System with Continuous-Valued Condition Attributes ...	108

Table 4.16. Partition Table for Cut Points Selection	109
Table 4.17. Partition Table for Cut Points Selection after First Iteration	109
Table 4.18. Partition Table for Cut Points Selection after Second Iteration.....	110
Table 4.19. The Information System after Discretisation.....	110
Table 4.20. An Information System with Continuous-Valued Attributes	112
Table 4.21. The Information System after Discretisation of the Decision	113
Table 5.1. An Information System with an Unrelated Condition Attribute, 'Colour' ..	121
Table 5.2. Rule set Derived from Table 5.1	121
Table 5.3. A More Comprehensive Information System.....	122
Table 5.4. Rule set Derived from Table 5.3	122
Table 5.5. An Extract of the Past Design Database	136
Table 5.6. Relative Discernibility Matrix for Decision Attribute F.....	137
Table 5.7. Relative Discernibility Matrices for Decision Attribute G	137
Table 5.8. Rule Set for Decision Attributes F and G	138
Table 5.9. A Modified Information System (Partial).....	138
Table 5.10. Updated Layer of the Global Discernibility Matrix for Decision Attribute D	139
Table 5.11. Updated Layer of the Global Discernibility Matrix for Decision Attribute E	140
Table 5.12. Temporary Relative Discernibility Matrix for Decision Attribute D	140
Table 5.13. Temporary Relative Discernibility Matrix for Decision Attribute E.....	141
Table 5.14. Updated Complete Rules based on Decision Attributes D AND E.....	141
Table 5.15. Functionalities of the Modules.....	148
Table 6.1. System Functions and Interface/Module Description.....	153

Table 6.2. A Summary of Elastomer Mold Design Variables	161
Table 6.3. Summary of Molding Design Synthesis	164
Table 6.4a. Case Study 2 Design Analysis Results for No Frill Customer.....	169
Table 6.4b. Case Study 2 Design Analysis Results for Fancy Customer	170
Table 6.4c. Case Study 2 Design Analysis Results for Pro Customer.....	171
Table 6.4d. Case Study 2 Design Analysis Results for Tech Customer	172

CHAPTER 1 INTRODUCTION

Chapter 1 Introduction

1.1. Background of the Research

1.1.1. Collaborative Design Trend

Product design is an integral part of product development and realisation. Essentially, during the process of product design, customer needs are transformed into a plan, and many decisions such as material selections, form creations and method adoptions need to be taken into consideration in order to realise a product. These decisions have a large impact on the final product, as about 80% of the product costs are committed during product design (Nevins and Whitley, 1989). During the process, new design concepts are often combined with or derived from well established past design concepts.

As technologies evolve, products have become more sophisticated. Multi-disciplinary knowledge is therefore needed to successfully design an acceptable product. To achieve this, recent trend has shown that many companies have started to specialise and then collaborate to share their expertise for the completion of design tasks. Accordingly, each company has a narrow but highly skilled expertise focusing on a specific area. Figure 1.1 illustrates the transformation from current design practices into collaborative design. From Figure 1.1, it appears that instead of working within a company to accomplish a design task, designers from different companies are working together. Additionally, stakeholders such as suppliers, manufacturers as well as customers are also involved throughout the design process. Other stakeholders such as the marketers and accountants may also need to be drawn in during the design process. The extensive interaction both between designers as well as between designers and stakeholders

CHAPTER 1 INTRODUCTION

causes an increase in the complexities of coordinating all the tasks. To support this extensive interaction, a suitable representation scheme for a design is needed. Ideally, this representation scheme is able to articulate the design itself as well as the reasoning behind the decisions taken during the design process.

Figure 1.1. Transformation from Traditional to Collaborative Design Practice

Collaborative design also causes design specifications that are largely fixed at the beginning of the conventional design process to evolve over time. This implies that the specifications are not anymore set in stone at the beginning of the design. They are subject to changes as the design progresses.

CHAPTER 1 INTRODUCTION

1.1.2. Collaborative Design in Conceptual Design

Typically, a design process can be divided into three stages (Hundal, 1997), namely conceptual, embodiment and detailed designs. The collaborative effort can occur in all the design stages. Figure 1.2 shows a simple schematic of a collaborative design system.

Figure 1.2. Collaborative Design in Design Process

Basically, the design process spans the three design stages, and stakeholders play an important role in assisting designers to complete a design. Stakeholder feedback can be solicited when it is needed. Stakeholders can also actively participate in the design process. As design reviews occur throughout the design process, design is actually an iterative process (Safoutin and Smith, 1996). Accordingly, as more information becomes available, design needs to be refined and improved. It should be emphasised that throughout the various design stages, collaboration across company boundaries are maintained.

Conceptual design has been established as the most important stage in design. In collaborative design, designers create, select, and develop design concepts to address

CHAPTER 1 INTRODUCTION

design problems. The creation of design concepts is based on the knowledge that the designers have accumulated through years of design experience. In general, the process of accumulating design knowledge or experience is realised incrementally. Basically, existing designs constitute a large amount of this knowledge, which can possibly be catalogued and extracted to help designers create new designs.

Additionally, it may not be known whether a design, after being manufactured, is able to meet the given specifications. This is especially true when the design is still in the earlier stage, i.e. conceptual design. Hence, early detection of inadequacy in a design can possibly help in reducing the iterations needed before the design is completed. If this is to be carried out in the conceptual design stage, a large amount of time and resources can possibly be saved since conceptual designs can be revised or even replaced when the actual resource committed is still minimal.

Early detection of design inadequacy can be done by estimating the performance of the new product during the conceptual design stage so as to derive maximum benefit. However, to estimate the performance of a design in the conceptual design stage is not a simple task. Basically, the conceptual design, which can be represented by a set of design concepts, is a collection of rough descriptions of the technology and procedures used, and the shape of the product (Ulrich and Eppinger, 2000). Thus, it is a difficult task to estimate the performance of a design based on the limited information. Simulation and product modelling are often used to check whether a design is able to meet design specifications. This is usually done in the detailed design stage, where the details of a product have been determined. As for the conceptual design stage, it does

not provide sufficient information for simulation as well as the necessary information for the construction and testing of a product.

1.1.3. Design Concept-Capability Mapping

The relationship between design concept and final design performance is depicted in Figure 1.3.

Figure 1.3. Design Concepts, Capabilities and Specifications Relationships (Alisantoso et al., 2004a)

As shown, it comprises three domains, namely design concepts domain, capabilities domain and specifications domain. Basically, the design concepts domain determines the capabilities domain through physical interactions of the design concepts. These interactions are realised after a product has actually been manufactured. The capabilities domain contains all the actual performance parameters that may or may not be required by the design specifications in the specifications domain. Capabilities that are required by a given set of design specifications are mapped in a one-to-one manner with corresponding specifications. On the other hand, those capabilities that are not required are simply not mapped. It should be added that even though these capabilities are not required, they may still exist. This is because they are the physical properties of the

CHAPTER 1 INTRODUCTION

product, which may or may not be important to the design. The mapping that governs the design concept-capability mapping is expected to be highly complex. Combinations of design concepts may generate different capabilities. Thus, a mapping mechanism that is able to manage this interaction is needed.

Rough set theory, which was advocated by Pawlak (1982), provides a novel approach to data analysis. It has been used in numerous domains (Pawlak, 1996a), including but not limited to, medical, economic, engineering and social science. One particularly interesting application of the rough set theory is in the domain of knowledge extraction. Here, rough set theory offers some practical advantages such as simplicity and efficiency as compared to other approaches (Pawlak, 1996b). Based on a set of data, it is able to extract inherent patterns, which are expressed as a set of rules. Accordingly, it is postulated that rough set theory can possibly be used to extract rules that govern design concept-capability mapping based on a set of past design data. It is further speculated that rough set theory can be extended to handle incremental updating of rules so as to mimic the way in which design knowledge is accumulated by a designer.

1.2. Objectives of the Research

The proposed research involves an in-depth investigation into collaborative conceptual design and related issues. The overall objectives of the work are to:

1. Establish a suitable representation scheme that is able to support communication among stakeholders efficiently,
2. Establish an approach for design concept-capability mapping so as to facilitate the evaluation of design capabilities of new products, and

CHAPTER 1 INTRODUCTION

3. Establish an approach for the generation of potential design concepts based on past designs.

Based on the above, it is envisaged that a prototype system can be realised to help designers in conceiving and selecting potential design concepts during the conceptual design stage. Specifically, the prototype system would provide assistance to designers working in a collaborative environment in two ways:

- Assessing the design capabilities of a new conceptual design, and
- Generating potential design concepts.

1.3. Scope of the Research

In order to achieve the objectives, the scope of the research comprises 4 major tasks:

Task 1: Establish the theoretical basis for the various approaches to design concept capability mapping

Basically, past designs of a family of products provide valuable information about design concepts and design capabilities that have been realised. Riding on the advent in artificial intelligence and information technology, this information can possibly be utilised for the evaluation of a new design of the same family of products. However, the information is expected to contain inconsistencies and variations, i.e. uncertain information. Thus, a technique that is able to handle uncertain information is necessary. In this connection, an in-depth study on such techniques as rough set theory, fuzzy set theory and neural networks will be performed. The study will help in establishing the theoretical basis for the realisation of a collection of approaches that can be used to perform design-concept capability mapping.

CHAPTER 1 INTRODUCTION

Task 2: Realise a novel knowledge representation scheme to facilitate Task 1.

Past design data is available in various modes and formats. Thus, it is necessary to establish an efficient knowledge representation scheme to facilitate the realisation of Task 1. It is envisaged that a pre-processor may be needed for the transformation of knowledge or information.

Task 3: Establish an approach to extract potential design concepts.

Based on the work established in Tasks 1 and 2, potential design concepts can possibly be extracted based on the design specifications provided by designers. This is a complex process, as it involves iterative actions, and review of design concepts, design capabilities and design specifications. For more complex cases, fine-tuning of design specifications is necessary. Hence, an in-depth study needs to be carried out in order to establish a metric to handle these issues, viz. iterative actions, review of design concepts, design capabilities and design specifications, and fine-tuning of design specifications.

Task 4: Develop and implement a prototype system for the approaches established in Tasks 1-3

A framework will first be developed to integrate the approaches established in Tasks 1-3. Subsequently, based on the framework, a prototype system will be realised. Suitable case studies will be performed to demonstrate the capability of the prototype system.

1.4. Organisation of the Thesis

The thesis is organised into 7 chapters. Chapter 2 gives a literature review. It discusses the various approaches to realise the tasks described in Section 1.3. It includes product

CHAPTER 1 INTRODUCTION

designs in general, collaborative design, artificial intelligence techniques and the rough set theory. Chapter 3 describes a novel representation scheme that is suitable for conceptual design in a collaborative environment. Chapter 4 presents a collection of novel approaches developed in this work for the handling of design concept-capability mapping. Based on the representation scheme and the various approaches described in Chapter 3 and 4, a framework of a prototype conceptual design assistant system for design analysis and design synthesis has been developed and described in Chapter 5. Chapter 6 presents the implementation of the prototype system. It includes two case studies that are used to illustrate the capabilities of the prototype system. Chapter 7 summarises the main conclusions reached in this work. It also outlines the recommendations for future work.

Chapter 2 Literature Review

2.1. Introduction

The increasing demand for better and cheaper products leads to the quest for even better designs as part of the overall product development strategy. Numerous attempts have been made to improve design processes. These attempts are facilitated by the introduction of new technologies such as the Internet and the improvement in existing technologies such as better computational power to facilitate the design process. This review aims at examining the trends and issues that are important to collaborative design. In the subsequent sections, general product design, collaborative design, conceptual design, design concept-capability mapping, and artificial intelligence approaches to design are addressed. An in-depth discussion will also be presented to provide the theoretical basis for the proposed research.

2.2. General Product Design

Product design is an integral part of manufacturing. It can be defined as the process of creating an artefact from ideas. As the definition suggests, product design covers all the activities from the identification of customer needs up to the creation of executable manufacturing plan for a new product. As such, the domain of general product design is indeed very broad. From a narrower perspective, design process can be divided into three stages (Hundal, 1997), namely conceptual design, embodiment design and detailed design. The output of one stage of a design process is the input of the subsequent stage. Accordingly, the input of the conceptual stage is product specifications that may be derived from customer needs. Product specifications serve as the basis of what a design

CHAPTER 2 LITERATURE REVIEW

is expected to achieve. Based on this input, designers generate product concepts that are assembled into one or more conceptual designs. Suitable conceptual designs are then used as inputs to the next design stage, which is embodiment design. In this stage, design concepts are developed further into products layout. The layout of a product can either be integrated or modular, and must be chosen carefully as it determines how the functionalities of the product are linked. It also affects the cost and application and thus the success of the product. The final design stage is detailed design. A fully detailed plan of the product is produced in this stage. It is important to note that there may be some pre- or post-stages that are done prior to or after the three design stages (Ulrich and Eppinger, 2000). Pre-stage may include the formation of design team and the identification of customer needs. Post-stage is done after the detailed design has been completed. It includes prototyping and production ramp up. These pre- and post-stages may not be directly linked to the design process for a product. However, they have a large bearing on the design process. For example, a product prototype can be realised and assessed, where applicable. The assessment determines whether the design should be manufactured.

The successive nature of the design process is not strictly followed since it is largely accepted that design is an iterative process. The decisions about a design are often made without complete information. Hence, designers have to base their decisions on the limited information currently possessed by them. As more information becomes available, design decisions that have been made earlier may need to be reviewed. Throughout the design process, the design is evaluated regularly and iteratively to check if it can proceed to subsequent stages. However, excessive iterations may cause costly

CHAPTER 2 LITERATURE REVIEW

delays, reworks and resource wastage. As a result, various approaches have been developed to reduce design lead-time by many researchers. One of these approaches concerns collaborative design, which is elaborated in Section 2.3.

2.3. Collaborative Design System

2.3.1. Introduction to Collaborative Design

Collaborative design is the process of designing a product through collective and joint efforts undertaken by a team of designers supported by other stakeholders (Shen, 2000). In working collaboratively, designers are not continuously accomplishing their tasks together. A large proportion of the tasks are, in fact, done individually (Liu et al., 2001). Designers are working individually to complete the actual design tasks while, at the same time, they are working together to plan, negotiate and evaluate design tasks. As already mentioned in Section 1.1.1, collaborative design provides an avenue to tackle the increasing complexities of products. In addition, there are also other reasons for the implementation of collaborative design besides the needs for different design expertise. For instance, a shorter product life cycle requires a shorter design time for a product to maximise its availability in the market. Collaborative design can help in achieving this by involving various stakeholders in the design process directly or indirectly. Moreover, globalisation brings about the needs to fulfil the requirements of customers in different geographical locations. Collaborative design is well poised to facilitate this.

Managing collaborative design across different geographical locations was not possible in the past. This has now been enabled largely by the progress in information technology. Current information technology has provided a more conducive

CHAPTER 2 LITERATURE REVIEW

environment by the use of the Internet and powerful computing tools. This development lends an encouraging prospect for collaborative design. Basically, the development of collaborative design technologies can be classified into three broad areas. They are:

1. Collaborative design infrastructure,
2. Virtual reality enhanced design environment, and
3. Intelligent tools for design.

These three areas focus on different aspects of collaborative design. Briefly, collaborative design infrastructure focuses on the creation of an information technology infrastructure that enables design to be done collaboratively. It forms the backbone of collaborative design systems. Virtual reality enhanced environment focuses on the usage of virtual reality as a tool for designers to work together. It enables designers to work together in real time and interactively with other stakeholders in examining and modifying designs. Intelligent tools for design explore the possibility of using intelligent tools to assist designers in design, particularly collaborative design.

From the review, it appears that collaborative design on a whole requires four basic functions:

1. Tracking and monitoring of decision process,
2. Anticipating problems and suggesting solutions,
3. Managing data sharing, and
4. Utilising external information sources.

These four basic functions complement each other to support a collaborative design environment. Tracking and monitoring of decision process is an important issue since

CHAPTER 2 LITERATURE REVIEW

collaborative design is likely to involve many stakeholders. These stakeholders may adopt different approaches in solving a design problem. Thus, an effective method that is able to track and monitor the decisions may become vital. Additionally, the processes and the considerations in arriving at a decision are crucial as they allow other stakeholders to understand the rationale behind the decision taken.

Anticipating problems and suggesting solutions are also important since the decisions taken by designers simultaneously may have or induce conflicts. These conflicts need to be detected and resolved for collaborative design to work seamlessly. The detection feature desirably needs to be reinforced with the ability to suggest solutions. The solutions provided by the system can then be used to help the designers resolve to the conflicts.

During the design process, the designers are likely to work with a large amount of data. The data can be in various modes and formats. Thus, it is necessary for the system to be able to handle various types of data so as to allow the designers to work together. Accordingly, effective data management is essential to provide all the designers with the information that they require.

Collaborative design aims at involving most of the stakeholders in a design process. However, not every stakeholder can or must be involved directly due to various reasons such as feasibility, necessity and security. Although direct involvement is not recommended, the information provided by these stakeholders may be important. By utilising external information sources, it is possible to exploit the information gleaned from these sources to enhance the design process.

CHAPTER 2 LITERATURE REVIEW

2.3.2. Collaborative Design System Functionalities

2.3.2.1. Tracking and Monitoring Decision Process

Decision-making in a collaborative design environment is interlinked. Essentially, the decision made by a designer may affect other designers. For example, the design concept selected by a designer may affect the choices available for another designer working on a related component. Thus, design decisions and design rationales need to be tracked and monitored to avert conflicts and misunderstanding.

In relation to this, Computer Aided Design (CAD) systems that are commonly used for communicating design drawings provide a primitive decision tracking and monitoring means through archiving design files. Essentially, these archived files can be retrieved so as to track the changes made by designers (Arbab and Wing, 1985; Arbab 1987). However, traditional CAD systems are intended to be operated by single user and thus are not naturally suitable or sufficient to accommodate simultaneous access of multiple-users. Zhou and Lin (1999) demonstrated an approach to implement cooperative geometry editing using traditional CAD systems. In their work, the decision making process was aided by a hybrid model, which was essentially derived by combining a replicated model and a centralised model. Using a replicated model, the CAD data model was copied to an individual machine used by a designer. As for a centralised model, the main model was kept in the main server. Further analysis showed that in a replicated model, maintaining consistency was a major problem while in a centralised model, heavy information traffic appeared to be the major drawback. To address these, in the hybrid model, design changes were transmitted and managed by the server. Thus,

CHAPTER 2 LITERATURE REVIEW

it could possibly reduce information traffic while maintaining consistency. However, it is important to note that such a model provides only a primitive form of decision tracking and monitoring.

To support collaborative design, more comprehensive functions are required to facilitate the interactions among designers. Song et al. (2001) proposed a collaborative design system for product design. The system focused on the exchange among designers in making design changes. As for managing interactions, conflict resolution, which was an important issue, was dealt with by utilising a series of so-called check-in, check-out and direct operations. Their collaborative design system had a three-layered mechanism comprising a user layer, a transactional layer and a data layer for data management. Such a mechanism allowed a user to create/edit a drawing while other users to participate passively or actively.

2.3.2.2. Anticipating Problems and Suggesting Solutions

Another crucial function that is needed for collaborative design is to anticipate problems and suggest solutions. While such a requirement also exists in traditional design practices, it is more complicated in a collaborative design environment, where a designer may make changes that are not in agreement with other designers. The designers may not be aware of the problems created in the initial stage, and find out only at later stages.

Dai and Drogemuller (1999) introduced a collaborative framework for design, which was applied to the domain of building designs. Their work aimed at automating the isolated and passive nature of building design services by using agent-based technology.

CHAPTER 2 LITERATURE REVIEW

It provided distributed interfaces for planners of a building design that were represented by agents. Decision support was performed in a CAD environment, which enabled designers to work collaboratively. The system aided the designers in checking for building regulations compliance and performing cost analysis. Whenever any of these constraints (regulations and/or costs) was violated, the designer would be alerted. This assistance, in effect, provided a limited means or avenue for anticipating problems. Specifically, the function possessed the ability to anticipate potential problems in meeting regulatory requirements as well as cost.

While the ability of a system to anticipate problems is a great help to designers, the problems have been left mostly to the designers. Finding the right solution to a problem is not a simple task, as it requires a close coordination with designers working on related components. Furthermore, finding a solution by making changes to the affected components may create additional problems to other components. In this respect, Shiau et al. (2000) implemented a multi-agent-based collaborative design environment for manufacturability assessment. Manufacturability assessment was carried out in the early design stage. A designer, through the design agent, could request a manufacturability assessment, which was transmitted to the manufacturing agent that generated a few process alternatives. The facility agent, which was attached to the equipment, was required to derive the alternatives. The alternatives were then evaluated by another agent and the results were sent to the designer who requested it. Based on the assessment, the designer could select the preferred design. Here, manufacturing availabilities and constraints were treated as the problems to be anticipated. Another example, CASE (Computer Aided Simultaneous Engineering) used multiple

CHAPTER 2 LITERATURE REVIEW

representations and agents to provide design and redesign suggestions (Taludkar et al. 1988). In so doing, the designer was able to anticipate the problems as well as to obtain advice in a timely manner.

2.3.2.3. Managing Data Sharing

Different types of data sharing management are available for collaborative design systems. Many of these are extensions of traditional data management and are meant for multi-users access. For example, Dai and Drogemuller (1999) used protocol and domain plan libraries for data management. These libraries were essentially traditional databases, which were linked together by an interface agent. The interface agent provided most of the data management functions. Other than using a custom-made database, popular databases such as Oracle could also be used (Sevy et al., 2000).

Another important aspect of data sharing management is the sharing of various design tools used to complete design tasks. Sevy et al. (2000) proposed a system known as Collaborative Design Studio, which was employed to enhance engineering design process through the integration of various off-the-shelf tools such as computer aided design and engineering tools, communication tools, and archiving functions. Lavana et al. (2000) suggested a collaborative design system known as OpenDesign, which provided an open user-configurable environment to support distributed collaborative design on the Internet. OpenDesign allowed a client to be configured for a specific project. The most affordable and suitable design tools could then be selected accordingly. An effective sequence of tasks was then configured to best suit a particular

CHAPTER 2 LITERATURE REVIEW

task. Data structure and revision control could also be customised. OpenDesign essentially tried to facilitate customisation so as to optimise the design process.

2.3.2.4. Utilising External Information Sources

The ability to gather and utilise information from external sources has not been addressed in a comprehensive manner. Collaborative design systems are usually constrained within closed boundaries of stakeholders directly involved in the design process. Macgregor et al. (2001) found that around 40% of design information was communicated among stakeholders who were not directly involved in a design process. Thus, constraining the information flow within those involved directly in the design will not be sufficient.

Hicks and Wiil (2001) developed an open hypermedia system to accommodate collaborative design. Basically, an open hypermedia system was able to provide an extensive computational environment beyond the standard hypermedia. Thus, it offered a greater potential for collaboration between stakeholders across different computational environments. Due to the nature of the open hypermedia environment, their system was able to extract information from external sources. However, the method used to manage information extraction had not been addressed. It is important to note that improper usage of external information sources may yield an avalanche of useless information. Similarly, Tian and Taylor (2001) implemented a web-based distributed collaborative design environment. The environment lacked the ability in managing the information obtained. Zhou et al. (2003) proposed an Internet based product design platform. In their work, a method to search for additional information through the Internet had been

CHAPTER 2 LITERATURE REVIEW

included. Unfortunately, it was unclear how the wealth of this information could be managed and stored.

2.3.3. Virtual Reality Enhanced Design Environment

Virtual reality was used in a number of prototype collaborative design systems. Basically, virtual reality offers a new approach to resolving the problem of having designers who are not geographically co-located. Using virtual reality, designers can work together as if they are in the same working space. For example, Zhang et al. (2000) presented a prototype system for virtual collaborative design. The system aimed at facilitating collaboration by providing a virtual environment for design, where designers could work on design tasks together. The prototype system used the Internet as a working platform from which the designers could be connected via a common server.

Immersive virtual reality systems were employed in a few prototype collaborative design systems. Bochenek and Ragusa (2001) suggested the application of virtual reality systems such as CAVE (Cave Automatic Virtual Environment) and Powerwall to collaborative design. In their work, a so-called Virtual Collaborative Design Environment was developed to improve the productivity of a design process. From the case study that was carried out on CAVE, it had been demonstrated that virtual reality was able to increase the comprehensibility of features of a design. In a different application domain, Frost and Warren (2000) used the CAVE system for architectural design. A group of users was involved in design of a laboratory. Their work revealed that virtual reality was indeed able to make the users experience the laboratory of the

CHAPTER 2 LITERATURE REVIEW

future. The problems concerning the design of the laboratory were identified during the virtual reality sessions. In other word, virtual reality was used to gather feedback from users after a conceptual design had been completed. However, further attempts to use the information gathered for decision-making was lacking.

2.4. Design Chain

Collaborative design is an important element for the formation of a design chain. A design chain is a link that spans the design process starting from its conception as ideas proposed by designers up to its utilisation as a physical product by customers. In contrast, conventional design practice usually considers a design process as terminated once a design plan has been finalised and submitted to the manufacturer for production. In this respect, design chain constitutes a new domain, where designs are performed in a cooperative but physically distributed fashion.

Figure 2.1. An electronic design chain (Martin and Schirrmeister, 2002)

Martin and Schirrmeister (2002) examined the design chain problem for the manufacturing of electronic products. They showed how the various designers of electronics components were linked in a design chain. Figure 2.1 shows a typical

CHAPTER 2 LITERATURE REVIEW

electronics design chain, where designers are working together to produce a complete product. Briefly, intellectual property (IP) block providers are used to design integrated circuits (ICs), operating systems, applications and middleware. Upon completion, those IP block providers that design ICs and operating systems advance their products to the semiconductor houses equipped with fabrication facilities. Subsequently, these semiconductor houses forward their design to the system houses. Semiconductor houses without fabrication facilities need to advance their design through pure-play foundries before going to system houses. On the other hand, IP block providers enable design application and middleware to send their design to system houses directly. System houses then assemble the design to be manufactured by original equipment manufacturers.

Basically, manufacturers can no longer afford to have their processes integrated vertically. They are expected to work closely with a network of partners. Thus, the aforementioned process is fairly similar to that of a supply chain. Furthermore, it is much more complex, as it has to handle design iterations and design revisions, which complicate the process. In reality, a more fragmented process that involves a number of companies is becoming more common. Such a trend is not only applicable to electronics products, but also other customer products due to the complexities in design.

The formation of a design chain brings about changes in the way in which a design process is managed (Otto and Wood, 2001). Basically, the collaboration of design tasks across company boundaries can cause numerous issues that have not been deemed crucial previously. Such issues as security and compatibility are becoming more important. As companies are working closely with each other, there are inherent

CHAPTER 2 LITERATURE REVIEW

security issues. Design chains are not static, but dynamic depending on product and market requirements. Thus, they open a security risk in relation to information sharing. Another important issue is the use of different platforms in different companies within a design chain. For companies to collaborate effectively, they need to adopt a common language. For this, STEP is an initiative to standardise the format of product data to facilitate information exchange between diverse systems (Arnold, 2000). However, much work still need to be carried out for STEP to be accepted by all stakeholders in a collaborative environment.

2.5. Conceptual Design Phase

2.5.1. Introduction to Conceptual Design Phase

During the conceptual design stage, many design concepts are conceived and evaluated. A set of design concepts typically forms a conceptual design (Ulrich and Eppinger, 2000). One or more conceptual designs are then selected to be refined further in the next stage. It has been established that 60% of the resources needed to develop a product are committed by the end of the conceptual design stage (Miller, 1993). This is in contrast with the fact that 80% of resources are committed by the end of the whole design process (Nevins and Whitley, 1989). Thus, conceptual design stage affects the design most as compared to other design stages. This is despite the fact that it is only the first stage of the overall product development process.

Figure 2.2 shows the flow of conceptual design. As shown, the input to the conceptual design stage is basically a set of customer needs, which are translated into a set of design specifications. These design specifications define the acceptable performance

CHAPTER 2 LITERATURE REVIEW

level to be achieved by a design. Next, the designers proceed to generate some conceptual designs using these design specifications. These conceptual designs are then evaluated so as to ascertain their potential in meeting the design specifications.

Figure 2.2. A Simplified Flow Chart for Conceptual Design

If a conceptual design is deemed to be acceptable, it is advanced to the next design stage. Otherwise, the conceptual design needs to be refined. It is also likely that, as the design evolves, the specifications may need to be modified due to, for example, the

CHAPTER 2 LITERATURE REVIEW

availability of promising new technologies. In other words, design specifications may vary during the course of conceptual design.

As already mentioned, the output of the conceptual design stage is a set of conceptual designs. Each of these conceptual designs may embody a set of design concepts. These conceptual designs are developed further in the later stages. At this stage, it is not certain whether they are able to meet the requirements stipulated in the design specifications. In practice, the designers can only guess or estimate the performance of the final products. If a conceptual design is not able to meet the specifications, then either the conceptual design needs to be revised or the design specifications have to be fine-tuned. Thus, design iterations are unavoidable.

It is likely that design iterations are carried out at much later stages during which design inadequacy can be properly detected through simulations and/or prototyping. This late detection and, thus, late modification and rework may result in more time and resources that are wasted. As such, an early estimation of the performance of the finalised product would enable the designers to detect whether the conceptual design selected are able to meet design specifications (Libardi et al., 1988; Shepard et al., 1988). Thus, modification to design should be done as early as possible so as to reduce design time and wastage of resources.

2.5.2. Design Knowledge Representation, Design Rationale and Design Purpose

Knowledge representation is an integral part of a design system. Essentially, an efficient and comprehensive representation scheme is required to support the communications among the stakeholders in order to realise a successful design. This is more so for a

CHAPTER 2 LITERATURE REVIEW

collaborative design environment where many stakeholders are involved in the design process. Design knowledge representation here refers to a scheme capable of capturing information that is produced and used during the design process.

As the detailed design of a product has not been fully finalised during the early stage of product development, i.e. conceptual design, modelling the exact geometrical shape of a product by CAD systems is not possible or meaningless. As a result, most designers attempt to focus on capturing product requirements for the generation of suitable design concepts at this stage of development. In this respect, techniques such as functional representation (Sembugamoorthy and Chandrasekaran, 1986 and Gero, 1990) can possibly be used to articulate the descriptions of a design concept. Functional representation allows designers to express the physical representation of a product, and the relationships between the intended functions of the product and the sequence of actions to realise them. In a collaborative environment for product development, rapid exchange of information on design concepts among the stakeholders in different geographical locations is inevitable.

Richter and Abowd (2000) defined design rationale as the explanations behind design. Accordingly, design rationale can also be used to denote the purpose of design, i.e. design purpose. Many approaches that attempt to formalise design rationale have been developed. Sections 2.5.2.1 and 2.5.2.2 outline two important concepts, namely Functional Representation and Design Rationale Representation.

CHAPTER 2 LITERATURE REVIEW

2.5.2.1. Functional Representation

In general, design information includes functions, physical behaviour and physical structures. Many design representation schemes are available. They include functional representation (Sembugamoorthy and Chandrasekaran, 1986), spatial and functional representation (Fenves and Baker, 1987), function-behaviour-state model (Umeda et al., 1996), function-behaviour-structure model (Qian and Gero, 1996), and function-environment-behaviour-structure (Deng et al., 2000).

It is important to note that the word, 'function', referred to in these representation schemes or models is not the same as a more commonly accepted definition coined by Pahl and Beitz (1988). Basically, Pahl and Beitz defined 'function' as the relationship between the input and output of materials, energy and information. Such a viewpoint is inherently limited, as it adopts an unnecessary classification that is unnatural. For example, a function that achieves its intended objective without using any input signal, energy and materials such as supporting static loading on a truss cannot be naturally represented. Accordingly, Umeda and Tomiyama (1997) argued that, in general, a function that does not transform something would not be satisfactorily represented. On the other hand, functional representation treats a function as a task that can or needs to be accomplished by a device. Such a definition offers a more flexible and natural avenue to represent function in conceptual design.

Functional representation (Sembugamoorthy and Chandrasekaran, 1986) models a device using three types of information, namely function, i.e. a description of the intended functions, structure, i.e. a description of parts and the relationships between

CHAPTER 2 LITERATURE REVIEW

parts, and process, i.e. a description of how the device achieves its functions. The function-behaviour-state representation (Umeda et al., 1996) denotes a device using three interrelated domains, namely function, i.e. a description of the intended function, behaviour, i.e. a description of the actions to achieve a function, and state, i.e. a description of the configuration of a structure. A similar representation scheme known as function-behaviour-structure was proposed by Qian and Gero (1995). Compared to the model proposed by Umeda et al. (1996), the definitions of the function and behaviour domains are alike while the structure domain corresponds to the state domain. Generally, Qian and Gero's representation scheme appears to be more formal compared to those of Umeda et al. and Sembugamoorthy and Chandraskaran.

Basically, function-behaviour-state or function-behaviour-structure (FBS) representation can be decomposed into two relationships, namely function-behaviour (F-B) and behaviour-structure (B-S) relationships respectively. The F-B relationship can be obtained by relating an action in the behaviour domain that realises a function perceived by designers. Such a relationship is largely subjective. In contrast, the B-S relationship is defined by the physical phenomenon where a state or structure is involved in an action stated in the behaviour domain, and is objective driven. These two models, i.e. function-behaviour-state and function-behaviour-structure, offer a functional representation scheme that is more systematic compared to that of Sembugamoorthy and Chandraskaran (1986). Deng et al. (2000) attempted to extend the FBS representation scheme. In their work, an additional domain, environment, was incorporated into the representation. Basically, the additional domain describes the

CHAPTER 2 LITERATURE REVIEW

environment where a product is supposed to work in. It postulates that environment plays an important role in achieving the intended function of a product.

2.5.2.2. Design Rationale Representation

As for design rationale representation, it includes issue based information system (IBIS) (Conklin and Burgess-Yakemovic, 1996; Nguyen and Swatman, 2000) questions, options and criterion (QOC) (Belotti, 1993; Karsenty, 1996), decision representation language (DRL) (Lee and Lai, 1991), and design pattern rationale graph (DPRG) (Baniassad et al., 2004). Basically, IBIS attempts to capture the design rationale when a design is being carried out. Design issues are then recorded and related to one another. Subsequently, arguments are used to support or reject corresponding issues. QOC describes design rationale using three parameters, namely question, option and criterion. Questions are answered using a number of options. These options are associated with the criteria, which are used to support or reject a design rationale. DRL are employed to capture design rationale using a complex network, which comprises a set of nodes and relationships. The nodes are used to depict decisions, problems, questions, alternatives and claims, while relationships are employed to denote 'supports', 'denials' and 'results'. DPRG makes explicit the relationships between design concepts in a design pattern, provides a graphical representation of the design pattern text, and supports the linking of those concepts.

Further analysis shows that the above-mentioned approaches possess different level of complexities in the order of QOC, IBIS, DPRG and DRL. However, the possibility of incorporating them into the FBS representation appears to be doubtful for they do not

CHAPTER 2 LITERATURE REVIEW

relate design rationale with the actual structure directly. Instead, they simply record the decisions taken during each design decision and the reasons for adopting such a decision or the reasons for not choosing certain design alternatives. In other words, although these approaches can be used to record the rationale behind design decisions, they only provide a vague description of the design itself.

2.6. Artificial Intelligence Approaches in Design

2.6.1. Intelligent Tools in Design

Intelligent tools such as concept generation, design evaluation, and conflict detection were developed to facilitate the completion of specific design tasks of a product. A vital task in design is the generation of design concepts. In this domain, Jiang et al. (2001) developed a collaborative design system known as CoDesign, which possessed a multi-agents based infrastructure for automated design of textile patterns. Basically, the agent based its suggestions on a knowledge base embedded in a pattern knowledge library. The pattern library also functioned as CoDesign's data management system, which was controlled by another agent. Although CoDesign had the potential to assist designers in carrying out design tasks, its application was limited to textile pattern design, which has the characteristics of having repetitive components, i.e. patterns. This limited its capability.

Another important application of intelligent tools is conflict detection. In the domain of product design, Xie (2001) proposed a collaborative design system known as Design Advisor. It was able to assist designers in identifying the impacts of design changes. Basically, when a design change took place, Design Advisor was able to inform the

CHAPTER 2 LITERATURE REVIEW

designer about the estimated overall impacts of the proposed change on the entire product. Other designers who were affected by the changes would also be informed. Thus, Design Advisor attempted to maintain the consistency of a distributed design automatically. However, this was limited to design changes that can be related explicitly to the product.

Huang et al. (2001) developed a system known as DARFAD to automate mechanical design. The system incorporated a model-based collaborative design facility that allowed designers and customers to take part in design activities via multi-agents over the Internet. It assisted the designers in creating alternative design concepts based on the design concepts submitted by the designers. However, its application domain was very specific since it focused only on mechanical designs.

2.6.2. Artificial Intelligence

As presented in Section 1.1.3, mapping of design concepts into design capabilities is a complicated process. In reality, experienced designers are able to conceive and/or select design concepts that are able to meet design specifications. This implies that past experience accumulated by the designers can help in accelerating this process.

Knowledge representation is an important domain in artificial intelligence (AI) research. It includes formal logic, semantic networks, frames and production rules (Turban, 1992). Briefly, using formal logic, a proposition can be divided into two parts, namely objects and assertions (Winston, 1977). Using objects, an assertion (conclusion) is made. A set of propositions constitutes the knowledge base. Inferences can then be made based on this set of propositions. Semantic networks store the knowledge as a set

CHAPTER 2 LITERATURE REVIEW

of networked nodes (Tuthill, 1990). Each node represents an object. Objects are interconnected by links that define their relationships. Frames represent all the information contained in objects (Edmonds, 1988). Each frame contains a set of slots that describes the objects. The frames possess a hierarchical structure that links the objects with specifically defined relationships. Production rules present knowledge in the form of IF-THEN rules (Newell and Simon, 1972). Typically, IF-THEN rules state ‘IF this premise is valid THEN this conclusion is taken’. This format provides an easy way to interpret information. As a result, the knowledge contained in the rules can be readily understood. Each of the knowledge representation modes has its advantages and disadvantages (Table 2.1).

Table 2.1. Advantages and Disadvantages of Different Knowledge Representation Schemes (Adapted from Turban, 1992)

Scheme	Advantages	Disadvantages
Formal logic	Facts are accumulated independently of use. Only valid assertions are made.	Detached representation and processing. Inefficient and slow with large data sets.
Semantic networks	Easy to grasp relationships.	Ambiguous meanings. Difficult in handling exceptions. Difficult to program.
Frames	Easy to set up. Comprehensive in expressing relationships.	Difficult to program. Difficult to inference.
Production rules	Simple syntax. Easy to understand. Simple inference. Highly modular. Easy to modify.	Difficult to follow hierarchy. Inefficient for large data sets. Difficult to represent structured knowledge

CHAPTER 2 LITERATURE REVIEW

In general, AI endows machines with the ability to carry out activities that are considered to require intelligence (Jackson, 1985). Thus, AI techniques can possibly be used to perform the aforementioned design concept-capability mapping. Some of the well-known AI techniques include knowledge based system, case based reasoning, artificial neural network, fuzzy set and rough set.

Basically, knowledge based system (KBS) (also known as expert system) is able to capture the knowledge possessed by domain experts (Savory, 1988). Through a question-and-answer session, novice designers can be guided to arrive at sensible design solutions. Case based reasoning (CBR) is a problem solving technology that aims at using past experience registered to solve current problems (Ritcher, 1998). Using such a technique, similar cases can be retrieved for the evaluation of design capability. Artificial neural network (ANN) is an approach that tries to mimic the way in which human brain works (Beale and Jackson, 1990). An ANN is composed of artificial neurons that are linked in a web-like structure. It is able to process information once the structure is established through a learning process. It can be used to complement other artificial intelligence techniques such as KBS (Medsker, 1992). Fuzzy set theory was proposed by Zadeh to handle real world phenomenon of approximate reasoning (Yamakawa, 1987). It provides the basis of a model for rational thinking in an imprecise environment. It has been used in a number of areas where imprecise boundaries of concepts are important. Fuzzy set theory provides a useful approach to treat non-quantitative data. It can be integrated with other techniques to realise a hybrid approach for problem solving. Rough set theory, which was proposed by Pawlak (1982), is a new mathematical tool to synthesise information from a set of data. A more in-depth

CHAPTER 2 LITERATURE REVIEW

discussion on the rough set theory is provided in Section 2.7. Further review on KBS, CBR and ANN can be found in the Sections 2.6.3, 2.6.4 and 2.6.5 respectively.

2.6.3. Knowledge Based System

Knowledge based systems (KBSs) aim at capturing the knowledge possessed by domain experts and storing it in a machine. The knowledge captured can be used to solve problems in the same domain. In the design, KBSs can be used to generate configurations of objects that can satisfy the constraints of a problem (Turban, 1992). They have been applied to many design areas. Lee et al. (1997) presented a KBS to assist designers in designing power converters. Optimal topologies could be generated based on the specifications given. Fung et al. (1988) developed a KBS for the design of an analogue IC design. The system was able to provide appropriate advice to improve IC designs. In their work, simulation was used for quantitative evaluation. Barnes et al. (1999) utilised a KBS to assist the design of an assembly. The system was combined with a geometric reasoning engine to provide early evaluation of a design. Helyacioglu and Insel (2000) proposed a KBS for container ship layout design. The system integrated empirical formulation and deterministic techniques to realise such a function.

KBS appears to dwell in a very specific and narrow domain from the above examples. The effectiveness of a KBS is only as good as the knowledge of the experts as it inherits the knowledge from the experts. The knowledge acquisition process is usually done with the help of knowledge engineers and is neither a simple nor inexpensive undertaking. Incompetence in the transfer of knowledge only aggravates the problems. It is important to note that maintaining a knowledge base can be a cumbersome task

CHAPTER 2 LITERATURE REVIEW

(Noelke, 1988). Moreover, with the advent in technology, the knowledge base that has been painstakingly developed can be out of date quickly.

2.6.4. Case Based Reasoning

The basic premise of case based reasoning (CBR) is that human perform decision making by retrieving past occurrences experienced by them (Riesback and Schank, 1989). In CBR, most decisions are not based on fundamental knowledge. Instead, past cases or experience are recalled and similar decisions are taken or adapted as appropriate. Thus, CBR can possibly be used in domains that heavily rely on past cases to derive the results. Some possible applications are law sentencing (Bain, 1989) and plan creation (Collins, 1989). However, the decisions generated in these areas are highly subjective and the merits of the decisions are difficult to assess.

The application of CBR has also been extended to experimental design support systems (Lee et al., 2000; Chiu, 2003; Wang and Liao, 2004). Basically, CBR assists designers by providing designers with past experience that may help in current situations. In this respect, Gomez de Silva Garza and Maher (1997) analysed a number of case based design systems deployed in different design domains, ranging from architecture and devices to software. It was found that in design, there were mainly two main applications of CBR. One of which focused on indexing and retrieving past designs. The other, which appeared to be more useful since it was able to assist designers directly, concerned the adaptation of past designs. However, many problems remain to be solved for non-routine design, where the adaptation of past designs is considerably more difficult. This was evidence in the work of Borner (1998), which focused on the

CHAPTER 2 LITERATURE REVIEW

applications of case based reasoning to non-routine design. Essentially, in non-routine design, the variables that are used in a design are not directly retrievable from an existing design (Coyne et al., 1987; Gero, 1990). This implies that the information needed for the adaptation of past cases may not generally be available. As a result, the search space that needs to be explored increases exponentially and this may result in unsuccessful adaptation.

2.6.5. Artificial Neural Network

Artificial neural network (ANN) has been widely used to solve engineering design problems. After the network has been configured, it is trained with a set of data before using. Kang and Yoon (1994) used ANN to solve simple truss design problems. Based on the geometry and loading on the structure, the ANN was able to derive the cross-sectional area of the trusses. Zhou et al. (1996) employed a simple ANN model to optimise the designs of chemical substances in order to meet a set of criteria. Arslan and Ince (1996) applied ANN to the design of concrete slabs. Salajegheh and Heidari (2005) utilised ANN to obtain the optimal structure, which could withstand earthquakes. From these examples, ANN appears to be a potentially useful tool for design concept-capability mapping.

Hara and Nakayama (1996) found that the accuracy of ANN improved as the amount of training data increased. However, this would adversely prolong the training time of the ANN. Fu (1994) observed that more training data were needed in order to use ANN to solve complex problems. Such observation may also have an impact on design concept-capability mapping, as large amount of training data may not be readily available.

CHAPTER 2 LITERATURE REVIEW

2.7. Rough Set Theory**2.7.1. Introduction to Rough Set Theory**

Rough set theory has been used successfully in many problem domains (Pawlak, 1996a). Notably, in the domain of decision support, rough set theory has been employed to extract decision rules from information systems. For example, several rough set based computer software systems had been developed for data analysis. These include ROSETTA (Øhrn, 1997), LERS (Grzymala-Busse, 1992), and INFER (Wong and Ziarko, 1987). Besides, rough set theory has been used in many practical domains. Kusiak et al. (2000) developed an autonomous decision support system based on the rough set theory for data mining. The system was tested using a set of medical data of patients with lungs abnormalities, with more than 90% accuracy. Nguyen et al. (2001) applied the rough set theory to generate rules for a medical knowledge based system. The rules obtained were examined by medical doctors and were found to be reasonable. Lee and Vachtsevanos (2002) introduced a technique based on the rough set theory for the identification of defects in the tail-light of cars in a production line automatically based on a set of measurements. In design, there have also been numerous applications of the rough set theory. Shan and Wang (2004) proposed a rough set based approach that was able to explore design space and find the optimal solution. Shi et al. (2003) combined rough set theory with fuzzy set theory to develop a system that was able to assist designers in detailed design of gating scheme for injection moulding. It appears that rough set theory can possibly be extended to handle design concept-capability mapping.

CHAPTER 2 LITERATURE REVIEW

Essentially, decision-making is a process where decisions are made based on a set of conditions. Analogous to such a notion, the proposed design concept-capability mapping aims at determining product capabilities (decisions) based on a set of design concepts (conditions).

Rough set theory offers significant advantages compared to other similar techniques (Pawlak, 1996b) as it provides an efficient algorithm to discover patterns in data and is able to evaluate the significance of data. Besides, it is able to produce results that can be interpreted directly.

Basically, rough set theory assumes that a set of object can be classified into overlapping subsets according to their attributes. Using rough set theory, a data set can be represented by a so-called information system. In the information system, an object can be fully represented by a set of attribute values. Mathematically,

$$I = (U, A) \quad \text{Definition 2.1}$$

where

I = information system,

U = universe (set of objects), and

A = set of attributes.

The classification of objects can be expressed as an indiscernibility relation. It can be expressed as

for $B \subset A$

$$\text{IND}_I(B) = \{(x, x') \in U^2 \mid \forall a \in B \ a(x) = a(x')\} \quad \text{Definition 2.2}$$

CHAPTER 2 LITERATURE REVIEW

Definition 2.2 indicates that a pair of objects, x and x' , are indiscernible by an attribute set, B , if the values of ALL the attributes in B for Objects x and x' are identical. This implies that if and only if the value of ANY of the attributes in B of x is different from that of x' , then the pair of objects is said to be discerned by B .

The attribute set of an information system can be categorised into two groups, namely condition and decision attributes. Thus, Definition 2.1 can be re-written as follows.

$$I = (U, (C, D))$$

Definition 2.3

where

I = information system,

U = universe (set of objects),

C = set of condition attributes, and

D = set of decision attributes.

The rules that govern the decisions in such an information system can be extracted using Definition 2.3. Mollestad and Komorowski (1999) derived decision rules using a so-called discernibility matrix that was defined based on the rough set theory. Basically, the discernibility matrix contains attributes that discern object-pairs. The entries in row i column j of the matrix are those attributes that discern Objects i and j . For clarity, the algorithm adopted by Mollestad and Komorowski is summarised as follows.

1. Based on the information system, form a decision relative discernibility matrix comprising n columns and n rows, where n is the number of objects.
2. Complete the $n \times n$ discernibility matrix. For each row and column (i, j) of the discernibility matrix, insert the condition attributes that discern Objects i and j

CHAPTER 2 LITERATURE REVIEW

where $i \leq n; j \leq n$, if the decision attribute for Objects i and j are different, i.e. discernable.

3. Form a discernibility function for each row. For each entry of a cell, combine the condition attributes registered using an OR Boolean operator to form a term. Subsequently, combine all the terms in a row with an AND Boolean operator.
4. Simplify the discernibility functions using Boolean algebra, if possible.
5. Fill the values of the condition and decision attributes for discernibility function i based on the values of the condition and decision variables of Object i to form Rule i .

Zhai et al. (2001) proposed another approach to the derivation of decision rules. In their work, a hybrid approach based on the rough set theory and genetic algorithms had been developed. More specifically, the data treated by the rough set theory was forwarded to a genetic algorithm based search engine for rule generation.

In the design concept-capability mapping, an information system, which is populated with past design objects, can be created. Here, design concepts and design capabilities are respectively represented by condition attributes and decision attributes of the information system. The rules derived from the information system can then be used to estimate the capabilities of new designs. However, the rough set theory cannot be applied directly to design concept-capability mapping, as there are four major issues that need to be tackled. These issues are incomplete information, non-discrete valued attributes, multiple decision attributes and incremental updating of rules.

CHAPTER 2 LITERATURE REVIEW

2.7.2. Incomplete Information

Analogy is frequently used to generate conceptual designs. Design concepts can be adapted from existing designs that are known to be able to perform the required function. However, existing designs may not be able to perform exactly what are required by the specifications of a new design. In addition, some of the attributes of existing designs may not be relevant to the new design. These discrepancies, in effect, create an incomplete information system. Review shows that rough set theory is able to handle incomplete information. Grzymala and Hu (2000) proposed a simple method, which was based on the probability, to repair an incomplete information system. In their work, a value with the highest probability of occurrence is assigned to the incomplete value. Such a method requires prior knowledge of data distribution, which may not be available for most data sets. Additionally, data with a small population may yield erroneous results, as the data distribution may not be able to be evaluated accurately.

Kryszkiewicz (1998, 1999) assigned a 'null' value to replace all the incomplete attribute values. The 'null' value represented all the possible values attainable by the attribute with incomplete value. In essence, the incomplete attribute value was assumed to be missing while in actuality it existed. The incompleteness is caused by error in data entry, for example. Thus, the original indiscernibility relation could be expanded, and represented by a tolerance relation known as similarity relation. Mathematically,

$$SIM_1(B) = \{(x, x') \in U^2 \mid \forall a \in B \ a(x) = a(x') \text{ or } a(x) = null \text{ or } a(x') = null\}$$

Definition 2.4

CHAPTER 2 LITERATURE REVIEW

Definition 2.4 indicates that a pair of Objects, x and x' , is considered similar in relation to a set of attributes, B , if they can possibly be indiscernible. Rules can then be derived from the incomplete information by replacing indiscernibility relation with similarity relation.

Similarly, Felix and Ushio (1999) solved the incomplete information problem by introducing rough discernibility relations, that is, surely discernible and possibly indiscernible. In their work, these relations were used for rules derivation in an incomplete information system to replace the original indiscernibility relation. Wang (2002) proposed a similar approach to handle the information by extending the operators of rough set theory. Different (and more complex) tolerance relations were introduced. These tolerance relations include non-symmetric similarity relation, valued tolerance relation and limited tolerance relation. It appears that limited tolerance similarity relation is superior to the rest. Interestingly, Pawlak (1998) cautioned users not to complicate the operations of rough set theory unnecessarily. This might lead to mathematical complications as well as philosophical questions caused by classifications without sharp boundaries.

Liang and Xu (2000) used information entropy theory to handle incomplete information. The incomplete attribute values were assumed to take any of the possible values. In this manner, the information system remained to be incomplete as no particular values were assigned to the attributes. Khoo and Zhai (2001) used a novel approach to deal with incomplete information. Their approach employed pseudo-objects to replace objects with incomplete information. After the information system had been repaired, standard rough set operations were performed.

CHAPTER 2 LITERATURE REVIEW

The abovementioned approaches are able to handle incomplete information. However, they are based on the same basic assumption that the values exist but not known. Thus, the incomplete information is due to the information has not been entered accurately. In other words, these approaches fail to solve those cases where incomplete information is due to irrelevant attributes, i.e. attributes may not exist for some of the objects at all. Such phenomenon is comparatively common in the domain of design. It provides the motivation to develop a new approach to manage this.

2.7.3. Non-Discrete Valued Attributes

Generally, continuous valued attributes are more common than discrete valued attributes. For a system to be practical and useful, it should be able to handle information system with continuous valued attributes. Many approaches had been developed to handle non-discrete valued attributes. Among them, using commonly accepted ranges of discrete values is, perhaps, the simplest. Its concept can be best illustrated by an example as follows.

It is assumed that the range of human body temperature can be discretised into three partitions, namely 'low', 'normal' and 'high'. Normal body temperature is one that is between 36.1°C and 37.8°C. As for low and high body temperatures, they are anything less than normal and anything higher than normal respectively. Although such a classification appears to be simple but meaningful, it requires prior knowledge about the commonly acceptable ranges.

In an earlier work, Quinlan (1986) proposed a learning algorithm known as ID3 (Iterative Dichotomiser). Basically, ID3 attempted to create a decision tree using a set of

CHAPTER 2 LITERATURE REVIEW

training instances and partition it by minimising the entropy. Hong and Tseng (1992) extended the ID3 algorithm and realised a so-called Generalised ID3. The Generalised ID3 offered more accurate classifications, simpler decision trees and shorter computing times. These had been made possible by generalising the formula used to minimise the entropy and were able to accommodate the availability of a priori domain knowledge. In other words, additional knowledge related to the domain can possibly be utilised. Quinlan further enhanced ID3 algorithm and coined the new algorithm C4.5 (Quinlan, 1993). Basically, C4.5 had better classification accuracy as compared to ID3. This, however, required a longer computation time. Subsequently, Alsabti et al. (1998), Gehrke et al. (1999) and Joshi et al. (1998) attempted to enhance C4.5 with the aim to improve its execution time and ability to handle large data sets. Quinlan (2000) further enhanced C4.5 to realise C5.0, which had better classification accuracy, faster and less memory requirements.

Nguyen (1997) developed an algorithm known as MD-Heuristic (Maximal Discernibility Heuristic) algorithm. MD-Heuristic aims at discretising continuous valued attributes with a set of semi-minimal partitions. Each cut point is chosen such that it maximises the number of objects that are discerned by the chosen cut point. This is done iteratively. The discretisation process is terminated when all objects that need to be discerned have been discerned. Alternatively, it is terminated when there is no more available possible cut points that can discern the remaining objects.

Komorowski et al. (1999) compared the performance of ID3, C4.5 and MD-Heuristic. It appears that MD-Heuristic is able to classify the object in the test sets more accurately

CHAPTER 2 LITERATURE REVIEW

as compared to others. In addition, shorter computing time was also reported. It is important to note that MD-Heuristic is also easier to understand.

The above-mentioned algorithms can possibly be adapted to discretise the attributes in the information system for the proposed design concept-capability mapping. Unfortunately, all of the approaches described thus far assume that the decision attributes are discrete valued. In relation to this, a new approach needs to be developed to handle continuous-valued attributes that may appear in both condition attributes as well as decision attributes for design concept-capability mapping.

2.7.4. Multiple Decision Attributes

The specifications of a design are likely to have more than one parameter. Thus, information systems with multiple decision attributes are to be expected in the mapping. Theoretically, rough set theory is able to handle information system with multiple decision attributes. Using the normal algorithm of rule generation for handling of multiple decision attributes, it is likely that the rules have to be completely regenerated from scratch every time there are changes in the design. Adding or removing of an attribute from the information system will invalidate the whole rule set. This is exacerbated by the iterative nature of design process. As a result, it may lead to unnecessary delay and can prevent the system to be effective.

A new approach needs to be developed to handle this problem. The approach should be able to simplify the rule generation process and accommodate the iterative nature of design. This can potentially make the system more efficient and thus practical.

CHAPTER 2 LITERATURE REVIEW

2.7.5. Incremental Updating of Rules

Rough set theory is well suited for design concept-capability mapping because it is able to generate reasonable results both for small data sets as well as large data sets. It is expected that more accurate results can possibly be obtained with a larger data set. However, rule generation is done in a way such that as the data set changes, the rule set needs to be updated so as to ensure its validity. Thus, it would be ideal if a mechanism for incremental updating of rules could be developed.

There are few algorithms that enable updating of rules to be done incrementally. Basically, as new objects are made available, they are appended to the information system. These newly appended objects modify the discernibility matrix and hence the rules need to be revised accordingly (Shan and Ziarko, 1995). Tong and An (2002) extended the above concept by incorporating additional algorithms to automatically update the parameters of the rules. These include coverage, confidence level and support as the rules are being updated incrementally. However, existing rough set based approaches to incremental updating of rules make use of all the attribute of all the objects present in the database to derive the rules. In the domain of design, this may not be the case, as not all the attributes are applicable to a certain design. In other words, only relevant objects and attributes from past design database need to be retrieved for rule generation. Thus, an approach that takes into account the unique requirements of design is required.

CHAPTER 2 LITERATURE REVIEW

2.8. Discussions

Collaborative design provides an avenue to improve the efficiency of design through enhancing the design process and reducing the lead-time of product development. Current research efforts in collaborative design are devoted to one or more of such functions as tracking and monitoring of decision process, anticipating problems and suggesting solutions, managing data sharing and utilising external information sources. The above review reveals that the work on collaborative design can be broadly categorised into three areas, namely collaborative design infrastructure, virtual reality enhanced design environment and intelligent tools for design. To recapitulate, collaborative design infrastructure provides an overall platform for the application of collaborative design systems. As for virtual reality enhanced design environment, it offers an avenue for stakeholders to communicate ideas and carry out design tasks interactively. This interaction is expected to yield a more productive design process. It has been found that attempt to utilise the above mentioned collaborative design infrastructure and virtual reality design environment to improve the process of design has been lacking. In other words, no single collaborative design system is yet able to address all the three areas. A more holistic approach that combines the strengths of the three areas, i.e. collaborative design infrastructure, virtual reality enhanced design environment and intelligent tools for design, is therefore desirable.

It is postulated that collaborative design will inevitably bring about the formation of a dynamic design chain. As already mentioned, the formation of a design chain requires a fundamental change in the way in which design tasks are carried out. More specifically, instead of having mostly a set of pre-determined design specifications of a product, it is

CHAPTER 2 LITERATURE REVIEW

anticipated that design specifications are likely to be dynamic and will probably evolve over time. As a result, the process of assessing design concepts may become more complex. As conceptual design is the most important stage in product design, an approach to facilitate the process of design assessment is therefore prudent and desirable. An early design concept-capability mapping that can predict the expected performance of a conceptual design is likely to be useful. It is envisaged that such a facility will enable designers to detect the inadequacy in the design of a product at the early stage of product development. This can possibly be realised using artificial intelligence techniques.

From the review, it is noted that collaborative design requires design information to be effectively communicated among the stakeholders. To realise this, techniques such as functional representations and design rationale representations may be able to provide a suitable framework for design knowledge representation. However, functional representations are not comprehensive enough to capture non-functional design purpose, which constitutes part of the essential information for the design of a product. On the other hand, design rationale representations seem to lack the ability in representing the actual physical design. Thus, to facilitate the exchange of information in a collaborative design environment, a more comprehensive design knowledge representation scheme is necessary.

In the domain of artificial intelligence, knowledge based system (KBS) provides an avenue to capture the design knowledge possessed by experienced designers. Two main problems, however, need to be resolved. First, it is necessary to address the difficulties in acquiring and capturing the knowledge needed for the establishment of an adequate

CHAPTER 2 LITERATURE REVIEW

and practical knowledge based system. Second, a novel approach to maintain and update the knowledge base is required (Noelke, 1988).

From the review, it appears that case base reasoning (CBR) can possibly be used for design concept-capability mapping. It is noted that CBR has been reasonably successful in dealing with domains such as law sentencing. In contrast with KBSs that treat decision making as a set of rules to be adhered to, CBR views it as a set of scripts, which are represented by past cases. Using CBR, relevant or similar past cases are retrieved and employed to reason about a new case. Such an approach appears to be promising, since in design concept-capability mapping, past designs can possibly be retrieved and formed the basis for the estimation of the capabilities of a new design. However, two fundamental issues in CBR, which concern case retrieval and case adaptation, need to be tackled. Case retrieval for design decision-making is inherently a difficult problem that invites much controversy. Basically, past cases are retrieved based on their similarities to the design of a product at hand. However, it is important to note that measurement of similarities in design is a complex issue to address. On the other hand, case adaptation is inevitable, as it is unlikely to have past design cases that are identical to the current one, i.e. new design. It is a well-known fact that in design, the number of possible alternatives is enormous. This implies that adapting past design cases to suit current specifications may not be easy. This is particularly so for non-routine designs where the number of possible alternatives is virtually unlimited.

As for artificial neural network (ANN), it is frequently used to complement other approaches. Fundamentally, ANN mimics the way in which human brain is perceived to be working. Information processing is expressed as a network of nodes. However,

CHAPTER 2 LITERATURE REVIEW

interpreting the information represented as an ANN cannot be carried out directly. Thus, it is difficult to ascertain whether the results generated by an ANN are reasonable or correct. Also in conceptual design, it cannot be guaranteed that there will be a large amount of data that can be used to sufficiently train the ANN.

It is noted that fuzzy set theory provides an approach to express non-quantitative attributes. Based on fuzzy set theory, an attribute can be represented by overlapping classes. The classification is expressed using membership functions, which are frequently defined based on intuition or subjective judgment. This, i.e. intuition or subjective judgment, may pose a fundamental problem to design concept-capability mapping.

Rough set theory provides a means to extract hidden patterns or decision rules from a set of data. This implies that the vague information inherent in conceptual designs can possibly be tackled using rough set theory. In other words, rough set theory offers good potential for the implementation of design concept-capability mapping. For this purpose, four issues associated with the application of rough set theory to design concept-capability mapping, namely incomplete information, non-discrete valued attributes, multiple decision attributes and incremental updating of rules, need to be addressed. Basically, incomplete information is likely to be encountered in design concept-capability mapping, as not all designs are identical. In practice, designers frequently use analogy to derive design ideas for a new design task. In this respect, design concepts of similar existing designs can possibly fulfil most of the specifications of the new design. As not all the attributes of similar existing designs are relevant, this implies that treatment of similar past designs needs to be carried out. Otherwise, this

CHAPTER 2 LITERATURE REVIEW

may limit the search space for possible design rules. For clarity, consider the design of a motorised light-weight two-wheeled vehicle. It is apparent that motorcycles and bicycles fit most of the product descriptions. As motorcycles and bicycles are not quite the same, some design concepts such as engine capacity that are important to a motorcycle may be irrelevant to the design of a bicycle. This results in the formation of an incomplete information system (Table 2.2).

Table 2.2. Sample Information System with Incomplete Information

Objects		Condition Attributes			Decision Attributes	
		Engine	Wheel	Material	Speed	Stability
Motorcycle 1	X ₁	small	small	heavy	slow	High
Motorcycle 2	X ₂	large	large	heavy	fast	High
Bicycle	X ₃		small	light		Low

The Condition Attribute Engine and the Decision Attribute Speed, are not relevant to the Object Bicycle. Thus, the Object Bicycle contains incomplete information. On the other hand, a motorcycle is much heavier than the bicycle. Thus, by removing either one of them, i.e. motorcycles or bicycles, the search space for the design rules of the motorised light-weight two-wheeled vehicles will be reduced correspondingly. It can be inferred from the above discussion that the quality as well as the number of design rules extracted will probably be limited. Conversely, it is postulated that by endowing the ability to handle incomplete information, useful rules for the design of innovative product can possibly be obtained.

CHAPTER 2 LITERATURE REVIEW

As mentioned, efforts to address the extraction of rules using incomplete information can be seen in the work of Grzymala and Hu (2000) and Kryszkiewicz (1998, 1999). These approaches, however, are not suitable for the proposed design concept-capability mapping, as the incomplete information, the engine capacity for bicycles for example, does not exist at all. This forms another class of incomplete information for an information system. Accordingly, a novel approach to handle this type of incomplete information in rule generation for design concept-capability mapping is needed.

Information systems with non-discrete attribute values, i.e. continuous valued attributes, cannot be tackled by the indiscernibility property of the rough set theory. To overcome this, the continuous valued attributes have to be partitioned or discretised (Komorowski et al., 1999). In design concept-capability mapping, both condition and decision attributes may register continuous values. For example, a design concept such as engine capacity, i.e. condition attribute, can be represented by an attribute having continuous value. Such a value can possibly be classified as 'small', 'medium', or 'large'. Similarly, a continuous valued design capability such as speed, i.e. decision attribute, can be classified as 'low' or 'high'.

Attempts to address non-discrete valued attributes using rough set theory can be seen in the work by Quinlan (1986, 1993, 2000) and Nguyen (1997). It is anticipated that some of these approaches can be adapted to deal with continuous valued attributes in design concept-capability mapping. Komorowski et al. (1999) had shown that the MD-Heuristic was superior to ID3 and C4.5. However, the MD-Heuristic algorithm can only handle discrete valued decision attributes. Hence, it has to be extended to deal with non-

CHAPTER 2 LITERATURE REVIEW

discrete valued decision attributes, as in design concept-capability mapping, decision attributes may be continuous valued.

2.9. Summary

Increasing product complexity, shorter product development life cycle and globalisation have resulted in a paradigm shift in product development, from single-organisation effort to multi-organisation collaborative effort, viz. collaborative design. Collaborative design can be viewed as the process of designing a product through collective and joint efforts undertaken by a team of stakeholders in different geographical locations. In this respect, a collaborative design environment requires some basic functions to make it more effective. These functions include tracking and monitoring of decisions, anticipating problems and suggesting solutions, managing shared data and utilising external information sources. In a collaborative design environment, the decision made by a designer may affect others. Thus, it is important for it to be tracked and monitored so as to ensure that the decisions made are appropriate. Furthermore, a decision made by a designer may contradict those made by other designers. In this respect, a facility to anticipate problems and suggest solutions is crucial to designers. Data sharing is also an important element to realise a collaborative design environment. Efficient data sharing helps designers to access vital information when needed. Since collaborative design invariably involves external entities, it is desirable to be able to utilise the information obtained from these sources. It has been established that work in collaborative design can be broadly categorised into three broad areas, namely collaborative design infrastructure, virtual reality enhanced design environment, and intelligent tools for design.

CHAPTER 2 LITERATURE REVIEW

As an emerging trend, companies begin to form design chains in order to stay competitive. This requires a more comprehensive design representation scheme to facilitate a more thorough communication among the stakeholders. It was revealed that functional representation schemes and design rationale representation schemes, although promising, are not adequate for this changing trend. Thus there is a need to devise a new conceptual design representation scheme to accommodate this.

Furthermore, the formation of design chain causes a shift in the way in which design specifications govern the design process. Design specifications are now dynamically revised as the design progress. Thus, it is necessary to detect design inadequacy as early as possible in view of the changing design specifications. To address this, the notion of design concept-capability mapping has been introduced. Basically, design concept-capability mapping aims at predicting the achievable product capabilities based on design concepts. It is proposed to be carried out in the conceptual design stage where early detection of design inadequacy is potentially most beneficial. In this respect, an intelligent tool for design can possibly be developed to assist designers in the design concept-capability mapping. It is envisaged that such an intelligent tool for design is able to assist designers in such tasks as predicting overall impact of design changes and creating alternative design concepts. In short, these areas of collaborative design infrastructure and virtual reality enhanced design environment are likely to be important for effective collaborative design. However, their potential advantages cannot be readily realised. They are highly dependent on the acceptance and the applications during the design process itself. Thus, their benefits cannot be easily quantified. In contrast, the potential benefits of intelligent tool for design can directly be realised.

CHAPTER 2 LITERATURE REVIEW

Artificial intelligence approaches such as KBS, CBR, ANN, fuzzy set theory and rough set theory can possibly be used to assist designers working in a collaborative design environment. It is a well-known fact that acquiring and capturing of expert knowledge in KBS is a cumbersome process. Thus, the maintenance and updating of knowledge base present a major problem for any KBS. As for CBR, effective case retrieval and case adaptation are two important issues that need to be tackled. To recapitulate, case retrieval is open to controversy, as the measure of similarity is difficult to define. Case adaptation has difficulties in handling non-routine designs. Although ANN can be integrated with techniques such as KBS, the results obtained are difficult to interpret or evaluate. Furthermore, in the domain of conceptual design, large amount of data for training an ANN may not be readily available. As for fuzzy set theory, fundamentally, its major weakness lies in the way in which membership functions are defined.

From the review, it appears that rough set theory can possibly be used to realise the proposed design concept-capability mapping. It has been established that direct application of rough set to design concept-capability mapping is hindered by four issues, namely handling incomplete information, treating of non-discrete valued attributes, managing multiple decision attributes and incremental updating of rules. Briefly, many researchers have attempted to handle incomplete information. However, none of the approaches addresses the situation, where the incomplete information is caused by some of the attributes are not relevant for some of the objects. Such a situation is peculiar in design where designers frequently use analogy to derive design ideas for a new design. This provides the motivation to develop a new approach to tackle the issues of incomplete information. Furthermore, the various approaches that

CHAPTER 2 LITERATURE REVIEW

were developed to handle non-discrete valued attributes assumed that decision attributes are totally discrete. A new approach to handle non-discrete valued decision attributes is thus needed. It is obvious that multiple decision attributes can naturally be tackled by rough set theory. However, using traditional approaches, rules need to be regenerated when there are changes in attributes. Rule generation, as a result, may become inefficient. An approach to address such an issue is therefore necessary. For practical purpose, as new objects become available, the rules need to be revised regularly. However, existing rough set based approaches to incremental updating of rules have limitations in dealing with information retrieved from past design database as not all the attributes is relevant to a particular design.

The review has identified the gaps in research associated with conceptual design. These gaps, which are addressed in Chapters 3, 4 and 5 include the following:

1. A novel design knowledge representation scheme for collaborative conceptual design (Chapter 3),
2. The three issues, viz. incomplete information, multiple decision attributes, non-discrete valued attributes, concerning the application of the rough set theory to realise the proposed design concept-capability mapping (Chapter 4), and
3. A rough set based approach to incremental updating of rules (Chapter 5).

The review has also laid the ground work for the realisation of a prototype conceptual design assistant for design analysis and design synthesis (Chapters 5 and 6).

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

Chapter 3 Purpose Behaviour Structure Representation

3.1. Introduction

Design is an integral component of product development, which includes manufacturing. In reality, in order to avoid costly reworks and delays, and to reduce the lead-time in product development, effective communication of design concepts among the various stakeholders such as customers, designers and engineers during the early stage of product development is of utmost importance. Accordingly, it is essential to establish an effective representation scheme to capture vital design knowledge during conceptual design. In a collaborative environment for product development, a more comprehensive approach that allows the stakeholders to communicate design concepts and requirements across company boundaries becomes necessary. As mentioned in Section 2.5.2, functional representation (Sembugamoorthy and Chandrasekaran, 1986; Gero, 1990) can possibly be used to express the descriptions of a design. However, functional representation alone is not comprehensive enough to capture all the vital design information that is necessary for the design process. In reality, it is common for a design process to be initiated by a set of design specifications. These design specifications can be translated into design purposes that include functional design purpose as well as non-functional design purpose.

Rosenman and Gero (1996) postulated design purpose as the reasons for designing a product or an artefact. In a broad sense, functions constitute only a part of design purpose, albeit an important one. Thus, to realise a more comprehensive representation scheme, there is a need to capture design purposes, both functional and non-functional,

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

for design conceptualisation. This chapter is organised as follows: Sections 3.2 highlights the application of functional representations to collaborative design. Sections 3.3 and 3.4 present the proposed Purpose-Behaviour-Structure (PBS) design representation scheme and a framework to utilise the proposed PBS representation scheme in a collaborative design environment respectively. A case study on the design of vacuum cleaners, which is used to illustrate the functionality of the proposed PBS representation scheme, is presented in Section 3.5. The last section, Section 3.6, summarises the main conclusions.

3.2. Functional Representation for Collaborative Product Development

Functional representation of design concepts in a collaborative environment needs to possess the following properties:

- The ability to support functional decomposition so as to enable design concepts with different levels of abstraction that are formulated by different stakeholders at different locations to be represented effectively, and
- Domain independence so as to allow cross-domain design analysis.

In collaborative product development, functional decomposition is an important process for it allows a complex design concept to be decomposed into the form of a hierarchical function tree. A comparison of the various approaches is depicted in Table 3.1. Table 3.1 compares 7 functional representation schemes. All of the schemes compared include Function domain and Structure domain. These domains are complemented by Behaviour domain, Layout domain, Specification domain and Environment domain, which differ from one scheme to another. It was noted that each scheme has its own

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

focus for its application. For example, Functional Representation (Sembugamoorthy and Chandrasekaran, 1986) focuses on Behaviour domain while SUPPORT (Nakatani et al., 1991) focuses on Function domain, Function-Structure relationship and Function-Specification relationship.

Table 3.1. Comparison of the Various Functional Representation Schemes

Approach	Domains	Focus	Abstraction Level	Domain Independence
Functional Representation (Sembugamoorthy and Chandrasekaran, 1986)	F, B, S	B	Yes	Yes
Function-Behaviour-State (Umeda et al., 1996)	F, B, S	F-B, B-S	Yes	No
Function-Behaviour-Structure (Qian and Gero, 1996)	F, B, S	F, B, S	Yes	Yes
Extended Function Logic (Sturges and Gero, 1996)	F, S	F	Yes	No
Integrated Function Behaviour Construction (Vargaz-Hernandez et al., 1996)	F, B, S, Layout	F, B, S	No	No
SUPPORT (Nakatani et al., 1991)	F, S, Spec	F, F-S, F-Spec	No	Yes
Function-Environment-Behaviour-Structure (Deng et al., 2000)	F, B, S, Env	B, S-Env	Yes	No

Note: F = Function; B = Behaviour; S = Structure; Spec = Specification; Env = Environment

It is apparent from Table 3.1 that functional representation and function-behaviour-structure are able to support functional decomposition and are domain independence. Essentially, functional representations attempt to capture the intended functions of a product.

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

However, some design purposes such as those that are defined as non-functional purpose in this work, i.e. *low power consumption* and *light-weight* cannot be naturally expressed as functions (Figure 3.1).

Figure 3.1. Functional Design Purpose and Non-Functional Design Purpose

For example, the filtration efficiency of a vacuum cleaner cannot be effectively represented by the function, *filter dust*, alone. As a result, an additional attribute needs to be attached to it so as to make the description more complete. Furthermore, non-functional requirements such as *low power consumption* can be represented by a functional design purpose with an attached attribute or by a non-functional design purpose. The choice of representation, i.e. functional design purpose or non-functional design purpose, is situation dependent. For example, if *low power consumption* has a

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

specific value that serves as a constraint, it may be more appropriate to represent it as a functional design purpose. The reason being *low power consumption* can be linked with the element in the behaviour domain. In so doing, the total power consumption can be made to meet the stipulated requirement. On the other hand, if *low power consumption* is used as a desirable feature of the design and no value being attached, then non-functional design purpose representation is more appropriate, as it can be added without further elaboration. There is a type of design purpose such as *light-weight* simply cannot be expressed as a functional design purpose. Although such a design purpose is non-functional, it is still important to the design of a vacuum cleaner. Thus, a more comprehensive representation that takes into consideration both functional and non-functional design purposes for conceptual design is necessary. Accordingly, in this chapter, a so-called purpose behaviour structure representation is proposed, and illustrated by a case study.

3.3. Purpose Behaviour Structure Representation

3.3.1. Basic Notions

As mentioned in Section 3.2, FBS provides a potential representation scheme to represent design concepts in a collaborative design environment. However, it lacks the ability in expressing non-functional design purpose. This section presents the work that attempts to extend the FBS representation by incorporating non-functional design purpose to realise a so-called Purpose-Behaviour-Structure (PBS) representation scheme for the capturing of design concepts (Alisantoso et al., 2005). The PBS representation scheme, which is in the form of a graph, is depicted in Figure 3.2. It

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

comprises three domains, namely a purpose domain, a behaviour domain and a structure domain respectively. The details of each of the domains are described as follows.

3.3.2. Structure Domain

The nodes in the structure domain on a PBS graph denote physical objects, which can be components, sub-assemblies or devices. They are known as structure elements and can be elaborated further by the attributes attached to them. These attributes can be used to describe the properties and features of the elements of the structure domain such as

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

type, material and shape. The edges on the PBS graph denote the assembly relationships between two structure elements such as an assembly and its sub-assemblies. The overall structure domain can be seen as a hierarchy of the product and can be decomposed into assemblies, sub-assemblies and components.

As more information becomes available when a design is being developed, a more detailed structure domain can then be constructed by adding and/or refining structural elements and attributes. By integrating the structure domain with the information extracted from catalogues and related CAD drawings, the entire design concept of a product can possibly be visualised.

The level of decomposition in the structure domain is largely based on subjective judgement, although the structure elements represent the physical objects of a design. As an example, in the development of the design concepts of an automobile, the structure domain may be decomposed into sub-assembly levels such as engine blocks, car doors and under carriage. Further decomposition into more detailed levels such as components level may not be necessary if the objective of the study or analysis is to understand the overall structure of the automobile or the working mechanism of its main components.

More formally, the structure domain can be represented by a graph, S , that consists of finite sets of elements as follows.

$$S = \{ S_E, S_A, S_V, S_R \}$$

where

S_E = a finite set of structure element nodes,

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

S_A = a finite set of structure attribute nodes,

S_V = a finite set of structure value nodes, and

S_R = a finite set of structure relation edges.

S_E denotes a set of assemblies or sub-assemblies that describes the design. The level of details is dependent upon the problem being solved. If a more detailed representation is needed, further decomposition is then necessary. S_A and S_V are sets of attribute-value pairs that describe the structure element nodes. For example, $S_A = \{Type, Material\}$ and $S_V = \{canister, polymer\}$. S_R is a set of relations that can be used to describe the relationship between pairs of element nodes such as $S_R = \{Type - canister, Material - polymer\}$.

3.3.3. Behaviour Domain

The nodes depicted in the behaviour domain on a PBS graph denote the spatial or the temporal states that the physical objects of the structure domain need to go through in order to achieve a required function. For example, in order for a vacuum cleaner to be able to perform the function, *suck dust*, the spatial or temporal states, *fan turns* and *air + dust flow*, in the behaviour domain can be associated. These spatial or temporal states are also known as spatial or temporal behaviour elements. Basically, spatial behaviour elements embody the behaviour of an object in the 2D or 3D space while temporal behaviour elements register the behaviour of an object within a certain time space. For example, the behaviour element, *fan turns*, can be considered as a temporal behaviour as the fan exhibits a behaviour that is to turn. On the other hand, the behaviour element, *air*

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

& *dust flow*, can be considered a spatial behaviour as air and dust travel from one location to another.

Additional information can be attached as attributes to these behaviour elements so as to provide further explanation. Accordingly, an edge in the behaviour domain represents the causal relation between different states, i.e. the effect of a state on the subsequent states. A behaviour element may be connected to one or more structure elements. For example, the behaviour element, *fan turns*, should be clearly connected to the structure element, *Fan*. On the other hand, a higher level behaviour element such as filtration mechanism should be connected to all the structure elements that are related to the mechanism. The construction of the behaviour domain is also based on subjective judgement, as different designers may have different perceptions about how a device works. The difference in perception, however, should not hinder its application to representing the design concepts of a product. Instead, the designers have to be more explicit in their description of how design concepts work when working on the behaviour domain so that it is sufficiently clear to be understood by others working on it.

More formally, the behaviour domain can be represented by a graph B which consists of finite sets of elements.

$$B = \{ B_E, B_A, B_V, B_R \}$$

where

B_E = a finite set of behaviour element nodes,

B_A = a finite set of behaviour attribute nodes,

B_V = a finite set of behaviour value nodes, and

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

B_R = a finite set of behaviour relation edges.

B_E denotes a set of states that describes how a design achieves its intended functions. For example, $B_E = \{fan\ turns, air\ \&\ dust\ flows\}$. B_A and B_V are sets of attribute-value pairs that describe the behaviour element nodes. For example, $B_A = \{speed, volumetric\ flow\}$ and $B_V = \{600\ rpm\}$. B_R is a set of relations to describe the relationship between pairs of element nodes. For example, $B_R = \{fan\ turns - air\ \&\ dust\ flows, fan\ turns - speed, speed - 600\ rpm, air\ \&\ dust\ flows - volumetric\ flow, volumetric\ flow - 2L/sec\}$.

3.3.4. Purpose Domain

The nodes depicted on the PBS graph in the purpose domain are used to register both the functional or non-functional design purposes. Compared to the FBS representation scheme, the PBS representation scheme is therefore more comprehensive and natural. For example, the functional design purpose, *clean flat surface*, describes the function which the intended product, *vacuum cleaner*, needs to fulfil. On the other hand, non-functional design purpose may include such requirements as colour, shape, weight and manufacturing constraints. The edges in the purpose domain on a PBS graph denote the decomposition of design purposes. Starting from the most basic, they are successively linked to higher levels of design purposes. For example, the functional design purposes, *remove dust* and *capture dust* are linked to the higher functional design purpose, *clean flat surface*. As for those edges linking the nodes of the behaviour domain with those of the purpose domain, they indicate a change in state in the behaviour domain in order to achieve certain design purpose in the purpose domain. For example, the behaviour element, *air & dust flows*, is to achieve the design purpose, *suck dust*. It is also possible

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

to have the edges connecting the nodes of the structure domain with those of the purpose domain directly, i.e. bypassing the behaviour domain. The connection can be achieved in two ways: 1) having a node of the purpose domain linked directly to a node of the structure domain, i.e. a structure element, if the existence of that structure element is directly related to the realisation of the design purpose; or 2) linking a node in the purpose domain to an attribute of a structure element in the structure domain. For example, the incorporation of a fuse is to prevent damage to the device due to short circuit. Thus, the purpose of *prevent damage due to short circuit* is linked directly to the *Fuse* node in the structure domain. On the other hand, the design purpose, *light-weight*, is linked to the attribute, *material*, since the choice of materials is likely to influence the achievement of the design purpose, *light-weight*. The feature or property described by the attribute facilitates the realisation of the design purpose. Similar to the structure and behaviour domains, the construction of the purpose domain is also based on subjective judgement. It depends on how the stakeholders perceive the purposes of a design concept.

More formally, the purpose domain can be represented by a graph, P , which consists of finite sets of elements.

$$P = \{ P_E, P_A, P_V, P_R \}$$

where

P_E = a finite set of purpose element nodes,

P_A = a finite set of purpose attribute nodes,

P_V = a finite set of purpose value nodes, and

P_R = a finite set of purpose relation edges.

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

P_E is a set that contains the intended purpose of a design. For example, $P_E = \{\text{clean flat surface, remove dust, capture dust}\}$. P_A and P_V are sets of attribute-value pairs that describe the element nodes of the purpose domain. For example, $P_A = \{\text{efficiency}\}$ and $P_V = \{95\%\}$. P_R is a set of relations to describe the relationship between pairs of element nodes. For example, $P_R = \{\text{clean flat surface - remove dust, clean flat surface - capture dust, capture dust - efficiency, efficiency - at least 95\%}\}$.

3.3.5. Discussion

In general, functional design purpose can be represented by a verb-noun pair (e.g. *clean surface, suck dust, capture dust*), while non-functional design purpose can be described using an adjective-noun pair (e.g. *light-weight, high durability*). A functional design purpose is connected to a physical object in the structure domain through a series of action objects in the behaviour domain. For example, the functional design purpose, *suck dust*, can be achieved by the behaviour elements, *fan turns* and *air & dust flows*. In contrast, a non-functional design purpose can be linked to the action objects in the behaviour domain to the structure elements in the structure domain. For example, the design purpose, *prevent damage due to short circuit*, is linked to the behaviour element, *circuit breaking*, and then the structure element, *Fuse*, in the structure domain. Alternatively, it can bypass the behaviour domain and connect directly to the structure elements in the structure domain. For example, non-functional design purpose, *light-weight*, is linked to the attribute, *material*, in the structure domain.

Consider a non-functional design purpose, *efficient energy consumption*, for the design of an air conditioning system (Figure 3.3). The non-functional design purpose is linked

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

to a mechanism of the air conditioning system called regeneration, which can be realised in two ways. First, as shown in Figure 3.3a, the description of the non-functional design purpose, *efficient energy consumption*, fits the above-mentioned adjective-noun pair format. Thus, efficient energy consumption is directly associated with the attribute *regeneration* of the air conditioning system in the structure domain. In other words, the regeneration mechanism is embedded in the air conditioning system in order to satisfy the non-functional design purpose, *efficient energy consumption*. Second, by rephrasing *efficient energy consumption* with *consume low amount of energy*, it is possible to use a functional design purpose, *consume energy*, in conjunction with an attribute, *low amount*, to represent the above-mentioned non-functional design purpose, *efficient energy consumption* (Figure 3.3b). Such a representation requires a mechanism in the behaviour domain to realise the functional design purpose, *consume energy*. The *Air condition system* in the structure domain is designed to fulfil the required functional design purpose, *consume energy*, with an attribute, *low amount*, tagging to it.

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

Figure 3.3. Different Ways to Represent a Design Purpose

This implies that, due to subjective judgement, the way in which a design purpose is described may result in different representations. In general, a rigid representation scheme that attempts to reduce or eliminate subjective judgement excessively may not be suitable for the description of design concepts, as it may limit designers' creativity. It is also less flexible. As a result, a rigid representation scheme may not be desirable due to its complexity. The same phenomenon can also be seen in the FBS (Qian and Gero, 1996) and the QOC (Belotti, 1993; Karsenty, 1996) representations. However, it is important to emphasise that, compared to the FBS and the QOC representations, the merit of the PBS representation scheme lies in its abilities in dealing with functional design purpose as well as non-functional design purpose.

Consider another non-functional design purpose, *low cost*, for the design of an air-conditioning system. The non-functional design purpose can be decomposed into *cheap maintenance* and *cheap hardware*, which follow the adjective-noun pair format

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

mentioned previously. Linking these to the corresponding structural elements will justify the selection of the design concepts, which are represented by the structural elements. Clearly, a simpler or more natural way to express them as functional design purpose is not possible. In this situation, the FBS and the QOC representation scheme cannot be used to represent it. The proposed PBS representation scheme, on the other hand, has no problem of capturing them.

Thus, the proposed PBS representation scheme for design concepts possesses the abilities to

- Capture both functional and non-functional design purposes,
- Support different levels of abstraction of design purpose, both functional and non-functional, through decomposition, and
- Provide a means to describe design concepts in a natural manner. Additional effort to document all the design decisions is not required.

From Table 3.2, PBS provides a substantial and practical extension to the FBS. It possesses all the capabilities of FBS. The weaknesses of the FBS in capturing non-functional design purposes and justifying design concepts have been overcome. These improvements enable PBS to support collaborative product development better. A PBS framework for collaborative product development is proposed and described in the next section, Section 3.4.

Table 3.2. Comparison between FBS and PBS

	FBS	PBS
Capture functional purpose	Yes	Yes

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

Capture non-functional purpose	No	Yes
Capture behaviour	Yes	Yes
Capture design concepts	Yes	Yes
Justify design concepts	No	Yes
Subjective level of abstraction	Yes	Yes

3.4. A PBS Based Framework for Collaborative Product Development

Calvanese et al. (1998) observed that schematic integration and data integration were two important approaches for the implementation of collaborative design. Schematic integration enables collaborating parties to access relevant data while maintaining different data formats. It also allows useful knowledge to be extracted from different sources. Further, it offers the benefit of allowing new comers to join the collaboration without much effort, as there is no need to re-format the data in order to work together. In reality, this is extremely difficult to achieve. On the other hand, data integration attempts to standardise the formats of data extracted from different sources. In so doing, design data can be accessed by collaborators or stakeholders at different locations to realise a seamless collaboration. As design concepts are inherently vague, the schematic integration approach may not be feasible. Thus, the data integration approach is adopted in this work for practical purpose.

Basically, collaborative design enables a complex design to be decomposed into manageable design tasks and handled by designers at different locations. When coupled

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

with a standard data format and the proposed PBS representation scheme for design, out-sourcing of design tasks and hence collaborative design become possible.

A PBS based framework for collaborative product development is depicted in Figure 3.4. As shown, the product requirements solicited are first analysed by the stakeholders. These requirements are then transformed into functional design propose or non-functional design purpose. Subsequently, using the proposed PBS representation scheme, the purpose domain is constructed. Basically, the construction of the purpose domain facilitates the decomposition of a product into modules. The designers can then proceed to work on different design purposes whenever possible to collaboratively derive the various mechanisms required by the behaviour domain and design concepts in the structure domain.

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

Figure 3.4. A PBS based Framework for Collaborative Product Development

Throughout the design process, the PBS representation provides a reference model and is refined as necessary. As an example, consider the functional design purpose, *capture dust*, of a vacuum cleaner (Figure 3.5).

Figure 3.5. Realising Design Purposes through Different Modules

It can be realised using two functions, namely *suck dust* and *capture dust*. These two functions can be achieved by two structure elements, a *Fan* to suck the dust and a *Filtration sub-assembly* to capture the dust. This implies that a vacuum cleaner comprises two main modules that need to be designed in order to realise the functional design purpose, *clean a surface*. These modules, once identified, can be designed by different designers at different geographic locations collaboratively. The inter-relationships between two modules are recorded to ensure seamless collaboration among affected designers. Other design purposes, functional or non-functional, can be similarly tackled. Once the design concepts have been finalised, the designers will then proceed to carry out detailed design. Subsequently, the manufacturing phase of product development will commence.

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

3.5. A Case Study

A case study on the design of a vacuum cleaner is used to illustrate the PBS representation scheme developed in this work. The PBS representation scheme is benchmarked against the FBS representation scheme in the case study (Figures 3.6 and 3.7).

The design of the vacuum cleaner consists of a *Power cord*, a *Switch*, a *Fan*, a *Dust bag*, a *Hose* and a *Brush* at component level and a *Filtration system* at sub-assembly level. The *Filtration system* can be further decomposed into a *Dust bag* and an *Air compartment* at component level. The components and the sub-assemblies belong to the structure domain.

The component *Switch* is linked to the behaviour *Toggle*. The behaviour *Toggle* is linked to the behaviour *Rotate*, which is also linked to the component *Fan*. Once the fan rotates, it will cause an air and dust to flow. This is represented by *Air+dust flow* behaviour. This behaviour is linked to the function *Suck dust*. It is also linked with behaviour *Air pass* and *Dust stop*. These behaviours, in turn, are linked with the function *Store dust*. The function *Suck dust* and *Store dust* are linked to a higher function *Capture dust*. This function, together with the function *Remove dust*, is linked with a higher function *Clean floor*.

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

FBS Model

Figure 3.6. A FBS Representation for a Vacuum Cleaner

As previously mentioned, the level of abstraction is based on subjective judgement. For example, the component, *Fan*, presents a design concept, which can be expressed as a fan sub-assembly. It, in turn, comprises a number of components. Different levels of abstraction according to the needs and focus of a design task can be applied. For the purpose of illustration, high level of abstraction, which is at sub-assembly level, is used in this case study. Thus, component *Fan* needs not be further decomposed into single-component level. Attributes are used to provide further details on some of these components. For example, component *Fan* comprises two attributes, namely *power consumption* and *type* respectively, and component *Brush* has an attribute, *type*.

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

Basically, FBS representation (Figure 3.6) alone is sufficient for designers to understand the basic function and the behaviour of the vacuum cleaner. However, it is important to note that the main task of the designers is to synthesise the design concepts of a vacuum cleaner collaboratively. Hence, the designers need to generate the design concepts based on the functions of the vacuum cleaner as well as other product requirements such as *light-weight* and *working condition*, which cannot be achieved by the FBS representation alone.

PBS Model

Figure 3.7. A PBS Representation for a Vacuum Cleaner

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

On the other hand, as shown in Figure 3.7, the PBS representation scheme is able to capture these product requirements, which can be described as non-functional design purposes such as the *temperature* requirement for the non-functional design purpose *working condition*. As for functional design purposes, for example, the *efficiency* of a vacuum cleaner is related to the function, *capture dust*. Having the design purposes, functional or non-functional, explicitly described and linked using the proposed PBS representation scheme, a comprehensive model for conceptual design can be realised. Thus, designers can easily refer to a product requirement or a specification that associates with a particular design concept on a PBS graph. This helps in shortening the lead-time in design.

In this case study, the desired minimum efficiency for capturing dust has been specified as a requirement. Hence, it will influence the design decisions for component design or component selection. Using the PBS representation scheme, components that are important to achieve this requirement can be traced to *Filtration system* and component *Fan*. Figure 3.7 shows two non-functional design purposes, namely *working condition* and *light-weight*. Basically, working condition such as *temperature* is an important constraint to be imposed on the design of vacuum cleaners. The likely mode of failure due to high temperature can possibly be attributed to the component, *Fan*, i.e. overheating of the motor. Thus, the non-functional design purpose, *working condition*, is linked to component *Fan* in the structure domain as shown in Figure 3.7. Correspondingly, *breakdown mechanism* must be included in the behaviour domain. Whether *breakdown mechanism* needs to be decomposed further is to be decided by the designers, i.e. based on subjective judgement. For non-critical products such as vacuum

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

cleaners, it may not be necessary to carry out further decomposition. On the other hand, for the design of critical products such as the jet engines of an aeroplane, *breakdown mechanism* needs to be scrutinised and further decomposed so as to ensure a safe and reliable design.

From Figure 3.7, it is apparent that the non-functional design purpose, *light-weight*, cannot be linked to the structure elements in the structure domain via the behaviour domain, as the product need not undergoing transformation in the behaviour domain to achieve *light-weight*. It bypasses the behaviour domain and is connected to the attribute of the corresponding structure elements in the structure domain directly. Here, the design purpose, *light-weight*, is linked to the attribute, *material*, of *Vacuum cleaner*. This implies that the non-functional design purpose, *light-weight*, can be achieved by choosing appropriate materials for the manufacture of vacuum cleaner. The case study also shows that the PBS representation scheme enables designers to include manufacturing constraints that are vital to the design process. It allows design purpose to be captured in a simple but effective manner by linking it with the design concepts.

3.6. Summary

A comprehensive design representation scheme known as Purpose-Behaviour-Structure (PBS) for collaborative product development has been proposed. The PBS representation scheme enables both functional and non-functional design purposes to be communicated among the stakeholders in a simple and yet effective way. Similar to functional representation schemes, the PBS representation scheme uses subjective judgement to decide different levels of abstraction. It has been established that the

CHAPTER 3 PURPOSE BEHAVIOUR STRUCTURE REPRESENTATION

perception of designers and the vague description of product requirements are likely to influence the way in which the PBS representation of a design concept is constructed. This, however, must not hinder the applications of the proposed representation scheme as long as the description is sufficiently clear for other stakeholders with similar level of knowledge to understand. Furthermore, the proposed PBS representation scheme provides a flexible format to describe design concepts. Thus, design purposes expressed using vague descriptions can be effectively represented. A PBS based framework for collaborative product development has been proposed. The framework enables designers at different geographic locations to collaborate and refine the design concepts of a product. A case study on a simple vacuum cleaner design was performed to illustrate the proposed PBS representation scheme. It has been shown that the PBS representation scheme is able to capture both functional and non-functional design purposes effectively. Using such a representation scheme, both functional and non-functional purposes could be traced to the corresponding structure elements or attributes. It has also been shown that product requirements can be easily included as design purposes using the proposed PBS representation scheme.

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

Chapter 4 Design Concept-Capability Mapping

4.1. Introduction

As discussed in Chapter 2, conceptual design is a complex and yet important process in product development. It requires multi-faceted knowledge from different disciplines in order to carry out a so-called design concept-capability mapping process, which can be used to estimate the expected performance of a product during conceptual design. In design concept-capability mapping, the conceptual design and performance of a product are respectively represented by a collection of design concepts and a set of design capabilities. For example, the design concepts of a hydraulic system can be obtained using different types of components such as actuators and valves, and different component layouts, while design capabilities can be specified using actuation speed and output power.

The design of a new product often relies on the knowledge and experience gained from past designs of similar products. Based on this understanding, it is postulated that design concept-capability mapping can possibly be derived using the information extracted from past designs. The mapping, which can be viewed as the transformation of design concepts to design capabilities, can be represented using different schemes such as formal logic, semantic networks, frames and production rules. As presented in Section 2.5.2, each of these representations has its advantages and drawbacks. In this work, the mapping is expressed using a set of design rules, i.e. production rules. Hence, given a set of design concepts, the set of design rules can be used to estimate the expected performances, i.e. design capabilities.

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

Basically, the information embedded in past designs can be organised as a set of attribute-value pairs. This implies that an object can be described using a set of attributes and each of these attributes has a value. In this way, a design concept can be represented by a set of condition attributes, and the associated design capabilities can be denoted by decision attributes (Figure 4.1).

Figure 4.1. Design Concept-Capability Mapping Representations

Figure 4.1 shows the process of design concept-capability mapping. As already mentioned in Section 2.7, the main problem associated with the generation of design rules for design concept-capability mapping using past design data lies in the existence of incomplete and inconsistent information in the information system. Basically, incomplete and inconsistent information can occur in the condition attribute as well as decision attributes of an information system. In this respect, rough set theory is able to provide a suitable metrics to handle design concept-capability mapping in the presence of incomplete and inconsistent information in the information system (Pawlak, 1996b). Using the rough set theory, design rules or the patterns of design decision-making can be discovered from an information system with incomplete information, i.e. incomplete

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

information system. In practice, the number of design specifications, i.e. decision attributes, is likely to be more than one and the attributes can be continuous valued. Thus, for design concept-capability mapping, it is also necessary to extend the rough set theory to handle issues concerning multiple decision attributes and continuous valued attributes.

This chapter aims at establishing the theoretical basis in order to address these issues. Section 4.2 presents a rough set based decomposition algorithm for the handling of multiple decision attributes. The treatment of incomplete information systems is reported in Section 4.3. An algorithm to handle non-discrete valued attributes is discussed in Section 4.4. The last section, Section 4.5, summarises the conclusions reached in this work.

4.2. Multiple Decision Attributes

4.2.1. Representation

A design can be viewed as an object, which is, in turn, denoted by a set of attributes with values, i.e. attribute-value pairs. Assume that a system (I) containing a set of objects $\{U\}$, each of which can be specified by a set of condition attributes $\{C\}$ and a set of decision attributes $\{D\}$ can be defined as

$$I = \{U, \{C, D\}\} \quad \text{Definition 4.1}$$

$\text{card}(D)$ is defined as the number of members of set D

where $\text{card}(D) > 1$ signifies a multiple decision attributes system, and $\text{card}(X)$ is the number of elements in the set, X .

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

An example of such a multiple decision attributes information system with 6 objects is depicted in Table 4.1. Each object has 5 attributes: 3 condition attributes and 2 decision attributes (i.e. $\text{card}(C)=3$ and $\text{card}(D)=2$). Accordingly, Condition Attribute A has 3 possible values while Condition Attributes B and C have 2 possible values each. Decision Attributes D and E have 3 and 2 possible values respectively.

The condition attributes represent those design concepts that have an influence on the design capabilities, which are represented by decision attributes. For example, condition attributes may represent a type of engine, a choice of material and a critical dimension of a design. As for condition attributes, they may represent the speed of a design and the stability of a design.

Table 4.1. An Information System with Multiple Decisions

Objects	Condition Attributes			Decision Attributes	
	A	B	C	D	E
X_1	1	1	1	1	1
X_2	1	1	2	2	1
X_3	2	2	1	3	1
X_4	2	1	2	1	1
X_5	3	1	1	2	2
X_6	3	1	1	3	2

As described in Chapter 2, indiscernibility is one of the important concepts in the rough set theory, which can be used to handle imprecise information. The indiscernibility relation can be expressed as

$$\text{IND}_I(B) = \{(x, x') \in U^2 \mid \forall a \in B \ a(x) = a(x')\}$$

Definition 4.2

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

where

B is the set of attributes that discerns the objects,

x and x' are a pair of objects within the universe U , and

$a(x)$ and $a(x')$ are the values of Attribute a of Object x and x' respectively.

The indiscernibility relation identifies those object-pairs that cannot be discerned based on the attributes. Hence, the indiscernibility relation with respect to decision attribute set D is defined as

$$\text{IND}_I(D) = \{(x, x') \in U^2 \mid \forall a \in D \ a(x) = a(x')\} \quad \text{Definition 4.3}$$

where

D is the set of decision attributes that discerns the objects,

x and x' are a pair of objects within the universe U , and

$a(x)$ and $a(x')$ are the values of Attribute a of Object x and x' respectively.

Based on Definition 4.3, the relative discernibility matrix for multiple decision attributes can be derived using the following principle (Komorowski et al. 1999),

'Two objects can be discerned if ANY of their decisions are not identical'

This implies that to discern a pair of objects, at least one of their decisions must be different. The above-mentioned principle, together with the algorithm proposed by Mollestad and Komorowski (1999), enable rules to be extracted from an information system (Section 2.7.1). It has been established that in design concept-capability mapping, the number of decision attributes is the same as the number of design specifications (Section 4.1). Accordingly, to generate the relative discernibility matrix

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

for decision attributes, every pair of objects has to be compared based on all the decisions, one at a time. However, such an approach may not be efficient for a relatively large information system with many attributes and objects. Additionally, if the design specifications have been modified due to changes in design requirements, the decision attributes need to be altered in order to match those in the modified design specifications. Consequently, the design rules have to be updated or regenerated.

4.2.2. Attributes Decomposition Algorithm for Multiple Decision Attributes

This section presents the proposed attributes decomposition algorithm (Alisantoso et al., 2004b) for the handling of multiple decision attributes. The proposed algorithm decomposes an information system with n decision attributes into n single decision attribute information systems. The design rules for each of these single-decision attribute information systems are then extracted. After which, the design rules extracted are combined. A formal proof of the attributes decomposition algorithm is given in Section 4.3.4. The attributes decomposition algorithm can be used to simplify the procedure for the handling of large information systems with multiple decision attributes. It is postulated that if the decision attribute set of an information system has been modified, only those rules that correspond to the change need to be regenerated.

Attributes Decomposition Algorithm

Given an information system consists of n objects and m decision variables,

1. Form m decision relative discernibility matrices, where m is the number of decision variables. Each matrix consists of n columns and n rows, where n is the number of objects. (Mollestad and Komorowski, 1999).

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

2. Complete the discernibility matrix. For the entry of Row i and Column j , enter the condition attributes that discern Objects i and j , if the decision for Objects i and j are different. Otherwise, enter Boolean value '1'. (See Section 4.2.4 for proof).
3. Form a discernibility function for each row of the discernibility matrix. For each entry in a row, combine the conditional variables using an OR Boolean operator to form a term. Combine all the terms in a row with an AND Boolean operator.
4. Simplify the discernibility functions, if possible.
5. Enter the values of the condition and decision attributes for discernibility function i to form Rule i . They can be extracted from the corresponding values of Object i .
6. Combine the rules using an AND operator.

A step-by-step illustration of the algorithm is as follows. From Table 4.1, two decision relative discernibility matrices (defined in Section 2.7.1) can be derived: one for Decision Attribute D and the other for Decision Attribute E. Tables 4.2 and 4.3 show the discernibility matrices for Decision Attributes D and E respectively. It is important to note that Table 4.2 was derived from Table 4.1 based on the value of Decision Attribute D without taking into consideration Decision Attribute E.

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

Table 4.2. Relative Discernibility Matrix for Decision Attribute D

	X ₁	X ₂	X ₃	X ₄	X ₅	X ₆
X ₁	1	C	AB	1	A	A
X ₂	C	1	ABC	A	1	AC
X ₃	AB	ABC	1	BC	AB	1
X ₄	1	A	BC	1	AC	AC
X ₅	A	1	AB	AC	1	1
X ₆	A	AC	1	AC	1	1

Table 4.3 Relative Discernibility Matrix for Decision Attribute E

	X ₁	X ₂	X ₃	X ₄	X ₅	X ₆
X ₁	1	1	1	1	A	A
X ₂	1	1	1	1	AC	AC
X ₃	1	1	1	1	AB	AB
X ₄	1	1	1	1	AC	AC
X ₅	A	AC	AB	AC	1	1
X ₆	A	AC	AB	AC	1	1

The alphabets A, B and C in Tables 4.2 and 4.3 are the attributes that can discern a pair of objects. For example, the entry of Cell X₁-X₂ (i.e. first row, second column entry) in Table 4.2 was obtained by comparing the attributes of Objects X₁ and X₂ shown in Table 4.1. Both objects have different values for Decision Attribute D (i.e. Decision Attribute D=1 for Object X₁ and Decision Attribute D=2 for Object X₂). A comparison of Condition Attributes A, B, and C for both objects shows that only Condition Attribute C can be used to discern these objects (i.e. Condition Attribute C=1 for Object

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

X_1 and Condition Attribute $C=2$ for Object X_2). Hence, Condition Attribute C is inserted into Cell X_1 - X_2 of Table 4.2. The entry of Cell X_1 - X_4 (i.e. first row, fourth column entry) of Table 4.2 can be obtained by comparing the values attained by the condition attributes of Objects X_1 and X_4 (Table 4.1). Both objects have the same values for Decision Attribute D (i.e. Decision Attribute $D=1$ for Objects X_1 and X_4). In this case, based on the attributes decomposition algorithm, '1' is entered into Cell X_1 - X_4 of Table 4.2. Entries of other cells in Table 4.2 can be similarly obtained. It was noted that the relative discernibility matrix is symmetrical with '1' in the diagonal cells.

Table 4.3 is derived using a similar algorithm from Table 4.1 based on the value of Decision Attribute E . Rules are then extracted from each relative discernibility matrix. As an example, the rules derived from the discernibility matrix of Decision Attribute D are tabulated in Table 4.4.

Table 4.4. Rules Based on Decision Attribute D

NO	IF	THEN
1	$(A=1)(C=1)$	$D=1$
2	$(A=1)(C=2)$	$D=2$
3	$((A=2)+(B=2))((B=2)+(C=1))$	$D=3$
4	$(A=2)((B=1)+(C=2))$	$D=1$
5	$(A=3)$	$D=2$
6	$(A=3)$	$D=3$

The first rule tabulated in Table 4.4 can be derived from the first row of the relative discernibility matrix of Decision Attribute D (Table 4.2) using the attribute values for

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

Object X_1 (Table 4.1). The attributes registered by the first row of Table 4.2 can be interpreted as a rule for Object X_1 .

If $C=1$ AND $(A=1$ OR $B=1)$ AND $A=1$ AND $A=1$ then $D=1$ Rule 4.1

Rule 4.1 can be simplified as:

If $C=1$ AND $A=1$ then $D=1$ Rule 4.2

Similarly, the fifth rule in Table 4.4 can be derived from the fifth row of the relative discernibility matrix of Decision Attribute D of Table 4.2 using the attributes values for Object X_5 shown in Table 4.1. The information registered by the fifth row of Table 4.2 can be interpreted as a rule for Object X_5 as follows:

If $A=3$ AND $(A=3$ OR $B=1)$ AND $(A=3$ OR $C=1)$ then $D=2$ Rule 4.3

Using Boolean operation, Rule 4.3 can be simplified as:

If $A=3$ then $D=2$ Rule 4.4

Similarly, the rules extracted from the relative discernibility matrix of Decision Attribute E are tabulated in Table 4.5.

Table 4.5. Rules Based on Decision Attribute E

NO	IF	THEN
1	$(A=1)(C=1)$	$E=1$
2	$(A=1)+(C=2)$	$E=1$
3	$(A=2)+(B=2)$	$E=1$
4	$(A=2)+(C=2)$	$E=1$
5	$(A=3)$	$E=2$
6	$(A=3)$	$E=2$

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

The rules for Decision Attributes D and E were combined using the AND Boolean operator. This is done by combining the same row of rules in Tables 4.4 and 4.5 to form a complete rule set as shown in Table 4.6.

For example, the second rules shown in Tables 4.4 and 4.5:

From Table 4.4: {If (A=1) AND (C=2) then (D=2)} Rule 4.5

From Table 4.5: {If (A=1) OR (C=2) then (E=1)} Rule 4.6

Rules 4.5 and 4.6 were combined using an AND operator to form

{If [(A=1) AND (C=2)] AND [(A=1) OR (C=2)] then (D=2) AND (E=1))} Rule 4.7

Rule 4.7 was simplified using the dual of Absorption Theorem to form:

{If (A=1) AND (C=2) then (D=2) AND (E=1)} Rule 4.8

The same treatment was applied to merge other rules from Tables 4.4 and 4.5. The results are tabulated in Table 4.6.

Table 4.6. Complete Rules based on Decision Attributes D AND E

NO	IF	THEN
1	(A=1)(C=1)	(D=1)(E=1)
2	(A=1)(C=2)	(D=2)(E=1)
3	((A=2)+(B=2))((B=2)+(C=1))	(D=3)(E=1)
4	(A=2)((B=1)+(C=2))	(D=1)(E=1)
5	(A=3)	(D=2)(E=2)
6	(A=3)	(D=3)(E=2)

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

4.2.4. A Proof of the Attributes Decomposition Algorithm

The attributes decomposition algorithm requires composite decision attributes to be decomposed into single decision attributes. Rules are then extracted for each constituent decision attribute based on the simplified information tables. Finally, the rules extracted from the simplified information tables are combined so as to derive the rules associated with the composite decision attributes. This section attempts to show that the rules derived directly from an information system with multiple decision attributes are the same as those derived using the attributes decomposition algorithm.

Given an information system $I = \{U, (C, D)\}$ comprising n number of objects where C and D denote the condition attributes and decision attributes sets respectively. A $n \times n$ relative discernibility matrix, $M_{[C, D]}$, for Decision Attribute D can be formed. The entry, $m_{[C, D]}$, in Row i and Column j is given by $m_{[C]}$, if the decision attributes of Objects X_i and X_j are not the same, otherwise the entry of $m_{[C, D]}$ is 1. Note that $m_{[C]}$ is the attributes that discerns Objects X_i and X_j .

More formally, the relative discernibility matrix for Decision Attribute D can be expressed as:

$$M_{[C, D]} = \{m_{[C, D]}(X_i, X_j) : 1 \leq i \leq n, 1 \leq j \leq n\} \quad \text{Definition 4.4}$$

where $m_{[C, D]}(X_i, X_j)$ is defined as:

$$m_{[C, D]}(X_i, X_j) = \begin{cases} m_{[C]}(X_i, X_j) & \text{if } D(X_i) \neq D(X_j) \\ 1 & \text{otherwise} \end{cases} \quad \text{for } 1 < i < n; 1 < j < n$$

where $m_{[C]}(X_i, X_j)$ is defined as:

$$m_{[C]}(X_i, X_j) = \{a \in C : a(X_i) \neq a(X_j)\} \quad \text{for } 1 < i < n; 1 < j < n$$

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

Based on the matrix established, a rule set, $R_{[C, D]}$, can be derived. Basically, a rule, $r_{[C, D]i}$, in a rule set is formed by first combining the entry in each of the cells of Row i into a term using the OR operators. Subsequently, these terms are joined using the AND operators to form a rule in accordance to the algorithm proposed by Mollestad and Komorowski (1999).

More formally, the rule set derived from the matrix can be expressed as

$$R_{[C, D]} = \{r_{[C, D]i} : 1 \leq i \leq n\} \quad \text{Definition 4.5}$$

where $r_{[C, D]i}$ is defined as:

$$r_{[C, D]i} = \left\{ \bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C, D]}(E_i, E_j) \right\} \Rightarrow D(E_i)$$

For a system with multiple decision attributes, Definition 4.4 can be extended to account for the basic notion that the values of the decision attributes of Objects X_i and X_j , are considered different if at least the value of one of the decision attributes set for the two objects are not the same. For an information system with q number of decision attributes, Definition 4.4 can be formally stated as:

for $1 < i < n; 1 < j < n; 1 < k < q$

$$m_{[C, D]}(X_i, X_j) = \begin{cases} m_{[C]}(X_i, X_j) & \text{if } \bigcup_{k \in \{1, \dots, q\}} d_k(X_i) \neq d_k(X_j) \\ 1 & \text{otherwise} \end{cases} \quad \text{Equation 4.1}$$

Similarly, Definition 4.5 can formally be expressed as

for $1 < i < n; 1 < j < n; 1 < k < q$

$$r_{[C, D]i} = \left\{ \bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C, D]}(X_i, X_j) \right\} \Rightarrow \bigcap_{k \in \{1, \dots, q\}} d_k(X_i) \quad \text{Equation 4.2}$$

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

For simplicity, let $D = \{d_1, d_2\}$. This implies that D comprises 2 decision attributes d_1 and d_2 . Hence, the rules derived from an information system with 2 decision attributes simultaneously can be expressed as follows.

$$m_{[C,D]}(X_i, X_j) = \begin{cases} m_{[C]}(X_i, X_j) & \text{if } [d_1(X_i) \neq d_1(X_j)] \text{ OR } [d_2(X_i) \neq d_2(X_j)] \\ 1 & \text{otherwise} \end{cases} \quad \text{Equation 4.3}$$

$$r_{[C,D]} = \left\{ \bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C,D]}(X_i, X_j) \right\} \Rightarrow d_1(X_i) \text{ AND } d_2(X_i) \quad \text{Equation 4.4}$$

Accordingly, based on the proposed attributes decomposition algorithm, the rules extracted from the information system with one decision attribute, d_1 , are given by:

$$m_{[C,d_1]}(X_i, X_j) = \begin{cases} m_{[C]}(X_i, X_j) & \text{if } d_1(X_i) \neq d_1(X_j) \\ 1 & \text{otherwise} \end{cases} \quad \text{Equation 4.5}$$

$$r_{[C,d_1]} = \left\{ \bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C,d_1]}(X_i, X_j) \right\} \Rightarrow d_1(X_i) \quad \text{Equation 4.6}$$

Here, to show that the rules derived from the information system with 2 decision attributes simultaneously and those derived using the proposed attributes decomposition algorithm are indeed the same, it is necessary to prove that the premise and the conclusion of the rules are the same, i.e.

$$r_{[C,d_1]} \cap r_{[C,d_2]} = r_{[C,D]}$$

The conclusions identical since for the rules derived using the attributes decomposition algorithm yields $d_1(X_i) \cap d_2(X_j)$ from Equation 4.6. While the rules generated by considering both d_1 and d_2 simultaneously yield $d_1(X_i) \text{ AND } d_2(X_j)$ from Equation 4.4. Thus, only the premises need to be examined further as follows:

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

$$\left[\bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C, d_1]}(X_i, X_j) \right] \cap \left[\bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C, d_2]}(X_i, X_j) \right] = \left[\bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C, D]}(X_i, X_j) \right]$$

Rearranging the left hand side of by pairing each $\bigcup \tilde{m}_{[C, d]}(X_i, X_j)$:

$$\begin{aligned} & \left[\bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C, d_1]}(X_i, X_j) \right] \cap \left[\bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C, d_2]}(X_i, X_j) \right] \\ &= \bigcap_{j \in \{1, \dots, n\}} \left[\bigcup \tilde{m}_{[C, d_1]}(X_i, X_j) \right] \cap \left[\bigcup \tilde{m}_{[C, d_2]}(X_i, X_j) \right] \end{aligned}$$

The possible pairing combinations are depicted in Table 4.7. For the rules derived simultaneously from the information system with 2 decision attributes, there are also 4 possible cases (Table 4.8).

Table 4.7. Possible Combinations of d_1 and d_2 for Rules Generated using the Attributes Decomposition Algorithm

$d_1(X_i) \neq d_1(X_j)$	$d_2(X_i) \neq d_2(X_j)$	$\bigcup \tilde{m}_{[C, d_1]}$	$\bigcup \tilde{m}_{[C, d_2]}$	$\bigcup \tilde{m}_{[C, d_1]} \cap \bigcup \tilde{m}_{[C, d_2]}$
Yes	Yes	$\bigcup \tilde{m}_{[C]}$	$\bigcup \tilde{m}_{[C]}$	$\bigcup \tilde{m}_{[C]}$
Yes	No	$\bigcup \tilde{m}_{[C]}$	1	$\bigcup \tilde{m}_{[C]}$
No	Yes	1	$\bigcup \tilde{m}_{[C]}$	$\bigcup \tilde{m}_{[C]}$
No	No	1	1	1

It is apparent that the right most column of Tables 4.7 and 4.8 are identical for all the possible cases.

$$\bigcup \tilde{m}_{[C, d_1]} \cap \bigcup \tilde{m}_{[C, d_2]} = \bigcup \tilde{m}_{[C, D]}$$

Since both the premise and the conclusion of the rules have been shown to be the same, it can be deduced that the rules derived by both algorithms are identical. It is important

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

to note that the proof is applicable to information systems with 2 decision attributes. It can be extended for information systems with more than 2 decision attributes.

Table 4.8. Possible Combinations of d_1 and d_2 for Rules Generated by Considering d_1 and d_2 Simultaneously

$d_1(X_i) \neq d_1(X_j)$	$d_2(X_i) \neq d_2(X_j)$	$\bigcup \tilde{m}_{[C,D]}$
Yes	Yes	$\bigcup \tilde{m}_{[C]}$
Yes	No	$\bigcup \tilde{m}_{[C]}$
No	Yes	$\bigcup \tilde{m}_{[C]}$
No	No	1

In general, consider an information system with q decision attributes, where $q > 2$, can be split into 2 information systems with $q-1$ and 1 decision attributes respectively. From Equations 4.1 and 4.2, for an information system with $q-1$ decision attribute, $m_{[C,D]}$ and $\eta_{[C,D]}_i$ can be expressed as follows,

for $1 < i < n; 1 < j < n; 1 < k < q-1$

$$m_{[C,D]}(X_i, X_j) = \begin{cases} m_{[C]}(X_i, X_j) & \text{if } \bigcup_{k \in \{1, \dots, q-1\}} d_k(X_i) \neq d_k(X_j) \\ 1 & \text{otherwise} \end{cases} \quad \text{Equation 4.7}$$

$$\eta_{[C,D]}_i = \left\{ \bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C,D]}(X_i, X_j) \right\} \Rightarrow \bigcap_{k \in \{1, \dots, q-1\}} d_k(X_i) \quad \text{Equation 4.8}$$

The above-mentioned process is carried out repeatedly until the information system is left with 2 decision attributes. Thus, the proof that was given earlier for an information system with 2 decision attributes can be applied.

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

4.3. Incomplete Information

4.3.1. Background of Incomplete Information

As discussed in Section 2.6.2, there are two main types of incomplete information in an information system, i.e. due to missing information and due to irrelevant information. In this work, incomplete information refers to the second type, which is a common phenomenon associated with the design of a new product based on past design experience. A novel algorithm has been developed to handle the second type of incomplete information.

4.3.2. Dissimilar Objects Algorithm for Incomplete Information

Table 4.9 shows an example of an incomplete information system. The information system consists of 5 objects with 5 attributes (3 condition attributes and 2 decision attributes).

Table 4.9. An Information System with Incomplete Information

Objects		Condition Attributes			Decision Attributes	
		Engine	Wheel	Material	Speed	Stability
Motorcycle A	X ₁	small	small	light	slow	high
Motorcycle B	X ₂	large	large	light	fast	high
Motorcycle C	X ₃	large	small	heavy	fast	low
Bicycle A	X ₄		small	light		high
Bicycle B	X ₅		small	heavy		low

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

Condition Attribute Engine of Objects X_4 and X_5 , i.e. Bicycles A and B, are unknown. This implies that these objects do not have the attribute, Engine. Similarly, Decision Attribute Speed for Objects X_4 and X_5 are also unknown.

Although Grzymala and Hu (2000), Kryszkiewicz (1998) and Jiye and Zongben (2000) attempted to develop techniques to handle information systems with incomplete information using rough set theory as previously discussed, none of them address the issue of incomplete information caused by irrelevant information. In this work, a novel algorithm known as dissimilar objects algorithm to deal with such an incomplete information system is proposed (Alisantoso et al., 2004a). The algorithm is summarised as follows:

1. Replace any incomplete information in the information system with 'UA' values, i.e. unavailable or irrelevant.
2. Derive the rules in the usual algorithm by taking the 'UA' as a new value for the attributes.
3. Simplify the rules by eliminating those rules containing 'UA'.

More specifically, consider the example depicted in Table 4.9 again. The blank cells in the table have been replaced by 'UA' to derive Table 4.10. Design rules are extracted by the proposed algorithm (Table 4.11). From Table 4.11, Rules 4 and 5, which contain 'UA', are eliminated, as they cannot be used for performance prediction.

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

Table 4.10. The Information System after Replacing Incomplete Information with ‘UA’

Objects		Condition Attributes			Decision Attributes	
		Engine	Wheel	Material	Speed	Stability
Motorcycle A	X ₁	small	small	light	slow	high
Motorcycle B	X ₂	large	large	light	fast	high
Motorcycle C	X ₃	large	small	heavy	fast	low
Bicycle A	X ₄	UA	small	light	UA	high
Bicycle B	X ₅	UA	small	heavy	UA	low

Table 4.11. Rules Derived from the Information System with Incomplete Information

NO	IF	THEN
1	(E=small)	(Spd=slow)(Sta=high)
2	((E=large)+(W=large))((W=large)+(M=light))	(Spd=fast)(Sta=high)
3	(E=large)((W=small)+(M=heavy))	(Spd=fast)(Sta=low)
4	(E=UA)(M=light)	(Spd=UA)(Sta=high)
5	(E=UA)(M=heavy)	(Spd=UA)(Sta=low)

Note: E: Engine; W: Wheel; M: Material; Spd: Speed; Sta: Stability

Table 4.12 shows the final rule set for design concept-capability mapping. The rules are checked against the original information system as shown in Table 4.9. It is obvious that Objects X₁, X₂ and X₃ satisfy Rules 1, 2 and 3 respectively. This showed that the rules are indeed reasonable since it was able to capture the pattern exhibited by the original information system.

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

Table 4.12. Final Rule Set after Elimination of ‘UA’

NO	IF	THEN
1	(E=small)	(Spd=slow)(Sta=high)
2	((E=large)+(W=large))((W=large)+(M=light))	(Spd=fast)(Sta=high)
3	(E=large)((W=small)+(M=heavy))	(Spd=fast)(Sta=low)

Note: E: Engine; W: Wheel; M: Material; Spd: Speed; Sta: Stability

4.3.3. A Proof for the Dissimilar Objects Algorithm

Consider an information system comprising $n+p$ objects. The first n objects do not contain any incomplete information, i.e. all the attributes of each object have corresponding values. The remaining p objects contain one or more attributes with incomplete information.

An information system with complete information

Consider a simplified information system comprising the first n objects. Mathematically, the information system can be expressed as $I = \{U, (C, D)\}$, where C and D denote the condition and decision attribute sets respectively. In this case, all the n objects do not contain irrelevant attributes (i.e. no unavailable information). Based on the approach proposed by Mollestad and Komorowski (1999), a $n \times n$ relative discernibility matrix, $M_{[C, D]}$, for Decision Attribute D can be formed. An entry in the matrix, $m_{[C, D]}$, in Row i and Column j is given by $m_{[C]}$, if the decision attributes of Objects X_i and X_j are not the same. Otherwise, a ‘1’ is assigned. Thus, $m_{[C]}$ is the condition attribute that discerns Objects X_i and X_j while the value ‘1’ represents the Boolean value ‘true’.

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

More formally, the relative discernibility matrix of decision attribute D can be expressed using Definition 4.4 as follows.

$$M_{[C,D]} = \{m_{[C,D]}(X_i, X_j) : 1 \leq i \leq n, 1 \leq j \leq n\}$$

where $m_{[C,D]}(X_i, X_j)$ is defined as:

$$m_{[C,D]}(X_i, X_j) = \begin{cases} m_{[C]}(X_i, X_j) & \text{if } D(X_i) \neq D(X_j) \\ 1 & \text{otherwise} \end{cases} \quad \text{for } 1 < i < n; 1 < j < n$$

where $m_{[C]}(X_i, X_j)$ is defined as:

$$m_{[C]}(X_i, X_j) = \{a \in C : a(X_i) \neq a(X_j)\} \quad \text{for } 1 < i < n; 1 < j < n$$

Based on the matrix established, a set of rules, $R_{[C,D]}$, can be derived. Basically, the proposition of a rule, $r_{[C,D]i}$, in the rule set is formed by first combining the entries in each of the cells of Row i into a term using the OR operator. Subsequently, these terms are joined using the AND operator to form a rule.

The rule set derived from the matrix can thus be expressed using Definition 4.5 as follows.

$$R_{[C,D]} = \{r_{[C,D]i} : 1 \leq i \leq n\}$$

where $r_{[C,D]i}$ is defined as:

$$r_{[C,D]i} = \left\{ \bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C,D]}(E_i, E_j) \right\} \Rightarrow D(E_i)$$

Incomplete information in an information system

Definitions 4.4 and 4.5 summarise the basic notions of the work by Mollestad and Komorowski, which was targeted at information systems with complete information.

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

For an ' $n+p$ '-objects information system with incomplete information, Definitions 4.4 and 4.5 can be extended and expressed as Equations 4.9 and 4.10.

$$M_{[C,D]} = \{m_{[C,D]}(X_i, X_j) : 1 \leq i \leq n+p, 1 \leq j \leq n+p\} \quad \text{Equation 4.9}$$

where $m_{[C,D]}(X_i, X_j)$ is defined as:

for $1 < i < n+p; 1 < j < n+p$

$$m_{[C,D]}(X_i, X_j) = \begin{cases} m_{[C]}(X_i, X_j) & \text{if } D(X_i) \neq D(X_j) \\ 1 & \text{otherwise} \end{cases}$$

where $m_{[C]}(X_i, X_j)$ is defined as:

$$m_{[C]}(X_i, X_j) = \{c \in C : c(X_i) \neq c(X_j)\} \quad \text{for } 1 < i < n+p; 1 < j < n+p$$

$$R_{[C,D]} = \{r_{[C,D]_i} : 1 \leq i \leq n+p\} \quad \text{Equation 4.10}$$

where $r_{[C,D]_i}$ is defined as:

$$r_{[C,D]_i} = \left\{ \bigcap_{j \in \{1, \dots, n+p\}} \bigcup \tilde{m}_{[C,D]}(X_i, X_j) \right\} \Rightarrow D(X_i)$$

Rules are generated by considering 'UA' as valid values of attributes according to the proposed dissimilar objects algorithm. In other words, Equations 4.9 and 4.10 have taken into account the additional objects with incomplete information. Accordingly, rules that contain 'UA' are eliminated from the rule set R , resulting in $n+q$ number of rules remaining in the rule set where $0 \leq q \leq p$. The extreme cases for q number of rules in the rule set are given below.

- $q = 0$ when all the rules corresponding to the objects with incomplete information are eliminated;
- $q = p$ when all the rules corresponding to the objects with incomplete information are retained.

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

There will be at least n number of rules remaining, as there are n objects with attributes having no 'UA' entry. Based on the proposed dissimilar objects algorithm, Equation 4.10 can be rewritten as follows after those rules with 'UA' values have been eliminated.

$$R_{[C,D]} = \{r_{[C,D]_i} : 1 \leq i \leq n+q\} \text{ for } 0 \leq q \leq p \quad \text{Equation 4.11}$$

where $r_{[C,D]_i}$ is defined as:

$$r_{[C,D]_i} = \left\{ \bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C,D]}(X_i, X_j) \right\} \cap \left\{ \bigcap_{j \in \{n+1, \dots, n+p\}} \bigcup \tilde{m}_{[C,D]}(X_i, X_j) \right\} \Rightarrow D(X_i)$$

Hence, the set of rules derived from the information system having p additional objects with incomplete information using Equation 4.11 have at least the same number of rules as those obtained previously using n objects with incomplete information (Equation 4.10).

The proposition of individual rules $r_{[C,D]_i}$ of Equation 4.11 is given by:

$$\left\{ \bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C,D]}(X_i, X_j) \right\} \cap \left\{ \bigcap_{j \in \{n+1, \dots, n+p\}} \bigcup \tilde{m}_{[C,D]}(X_i, X_j) \right\} \quad \text{Expression 4.1}$$

It is important to note that the first term of Expression 4.1, $\left\{ \bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C,D]}(X_i, X_j) \right\}$, is identical to the proposition of Definition 4.5. As the first and second terms of Expression 4.1 are joined using an AND operator, it can be deduced that the proposition of Equation 4.11 is indeed a subset of that of Definition 4.5. Hence, the rules derived from such an information system with objects having incomplete information based on the proposed dissimilar objects algorithm is more stringent compared to those obtained by the algorithm proposed by Mollestad and Komorowski, where incomplete information cannot be included. Accordingly, it is likely to be more specific and

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

potentially more useful for the evaluation of design capability. In addition, more rules are likely to be derived. As a result, additional and potentially useful rules can possibly be obtained. The reason being by including objects with some incomplete information would result in a larger search space for possible solutions to a conceptual design. A larger search space, in turn, is likely to provide a means to search ‘outside the box’. This so-called ‘outside the box’ search would result in the generation of more creative solutions, and possibly, a more robust design concept analysis.

4.3.4. Discussion on the Dissimilar Objects Algorithm

It can be argued that since the rules containing the variable, ‘UA’, have been eliminated from the rule set, those objects containing incomplete information such as Objects X₄ and X₅ in Table 4.9 should not be considered for the generation of design rules. In so doing, computation effort can possibly be reduced. Further analysis is carried out to examine such a proposition.

Table 4.13 shows the information system after eliminating objects with incomplete information (Table 4.9). Rules were extracted from the simplified information system (Table 4.14). A total of three rules would be extracted.

Table 4.13. Information System after Eliminating ‘UA’

Objects		Condition Attributes			Decision Attributes	
		Engine	Wheel	Material	Speed	Stability
Motorcycle A	X ₁	small	small	light	slow	high
Motorcycle B	X ₂	large	large	light	fast	high
Motorcycle C	X ₃	large	small	heavy	fast	low

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

Table 4.14. Rule set Derived from the Simplified Information System

NO	IF	THEN
1	((E=small)+(W=small))((E=small)+(M=light))	(Spd=slow)(Sta=high)
2	((E=large)+(W=large))((E=large)+(M=light))	(Spd=fast)(Sta=high)
3	((E=large)+(W=heavy))((E=small)+(M=heavy))	(Spd=fast)(Sta=low)

Note: E: Engine; W: Wheel; M: Material; Spd: Speed; Sta: Stability

It is apparent that Rules 1 and 3 of Table 4.14 are different from those derived by the proposed algorithm (Table 4.12). More specifically,

Rule 1 in Table 4.12

If (E=small) then (Spd=slow) AND (Sta=high) Rule 4.9

Rule 1 in Table 4.14

If [(E=small) OR (W=small)] AND [(E=small) OR (M=light)] then (Spd=slow)
AND (Sta=high) Rule 4.10

It is apparent that Rules 4.9 and 4.10 draw the same conclusion. Thus, only the premise needs to be examined. The premise of Rule 4.9, ‘[(E=small) OR (W=small)] AND [(E=small) OR (M=light)]’, can be further simplified as ‘(E=small) OR [(W=small) AND (M=light)]’. Compared with that of Rule 4.10, viz. ‘(E=small)’, it is apparent that the premise of Rule 4.9 is a subset of that of Rule 4.10. This signifies that the premise of Rule 4.9 is more stringent. Hence, the proposed algorithm (by including objects that are dissimilar) has improved the preciseness of the mapping. Similarly,

Rule 3 in Table 4.12

If (E=large) AND [(W=small) OR (M=heavy)] then (Spd=fast) AND (Sta=low) Rule 4.11

Rule 3 in Table 4.14

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

If [(E=large) OR (M=heavy)] AND [(W=small) OR (M=heavy)] then (Spd=fast) and (Sta=low) Rule 4.12

It is obvious that the premise of Rule 4.12 can be transformed into '{(E=large) AND [(W=small) OR (M=heavy)]}' OR '{(M=heavy) AND [(W=small) OR (M=heavy)]}'. This shows that the premise of Rule 4.11 is a subset of that of Rule 4.12.

It is important to note that while it is beneficial to append similar but not exactly the same objects to an information system, the selection of objects should be done cautiously, as the inclusion of objects with large diversity may yield unreasonable results. Additionally, the values assigned to the attributes of an information system need to be consistent. In design, attribute values such as 'high', 'low' and 'heavy' are highly dependent on the product. A 'high' for an attribute of a product maybe considered a 'low' for another. For example, a bicycle with a weight of 40 kg can be considered as 'heavy' while in the design of a motorcycle, it can be considered as 'light'. An approach to handle this will need to be considered.

4.4. Non-Discrete Valued Attributes

4.4.1. MD-Heuristic Algorithm

As briefly mentioned in Section 2.6.3, the MD-Heuristic algorithm (Nguyen, 1997) works by successively finding a cut point that can discern the highest number of object-pairs in an information table. For clarity, the algorithm is outlined as follows.

1. Construct a list of possible cut points for each condition attributes. The possible cut points are the mid points of successive values of each attribute.

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

2. Construct a partition table from the information system. The rows contain all the object-pairs that need to be discerned. The columns register all the possible cut points.
3. Complete the table using the following guidelines. For an entry with row (u_x, u_y) and Column c_i , enter '1' if Cut point c_i can discern u_x and u_y . Otherwise, enter '0'.
4. Count the number of '1' in each column. Note that a '1' is assigned if the cut point is able to discern an object-pair. This determines the number of object-pairs to be discerned by the cut point in the column.
5. The possible cut point corresponding to the column with the highest number of entry, '1', is then selected as a cut point.
6. Remove those rows with '1' in the column that was selected in Step 5. This will remove object-pairs that have been discerned by the selected cut point from further treatment.
7. If there are object-pairs (represented by rows) to be discerned and there are possible cut points that can discern objects-pairs, repeat Step 4. Otherwise, terminate the algorithm.

Table 4.15 shows an information system with 6 objects and 3 attributes (2 condition attributes and 1 decision attributes). Both condition attributes are continuous-valued and have to be discretised.

The values for each variable to be discretised are first sorted in ascending order. The mid-point of successive values of each attribute is taken as a possible cut point. For example, the values of Condition Attribute A in Table 4.15 can be sorted in the order of '0.4, 0.7, 0.8, 1.1'. The mid-point of the successive value 0.4 and 0.7 is 0.5. The other

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

midpoints are 0.75 and 0.95 respectively. Together, they form the sequence, ‘0.5, 0.75, 0.95’. These are the possible cut points for Condition Attribute A. The cut points for Condition Attribute B can be derived in a similar fashion to yield the sequence, ‘1.45, 1.85’. Using these cut points, a partition table can be formed. For example, Table 4.16 shows the partition table for the information system depicted in Table 4.15.

Table 4.15. An Information System with Continuous-Valued Condition Attributes

Objects	Condition Attributes		Decision Attribute
	A	B	D
X ₁	1.1	0.9	Yes
X ₂	0.8	1.9	Yes
X ₃	0.4	1.9	No
X ₄	0.7	1.8	No
X ₅	0.8	0.9	No
X ₆	1.1	1.8	Yes

A ‘1’ in the partition table means that the cut point is able to discern the object-pair. For example, the entry in row 1 of column 1 is a ‘1’. This implies that the cut point at A=0.5 is able to discern Objects X₁ and X₃, based on the data registered in Table 4.15. On the other hand, the cut-point, A=0.5, is not able to discern Objects X₁ and X₄, based on the data recorded in Table 4.15. Hence, the entry in row 2 of column 1 is a ‘0’.

The total number of ‘1’ in each column was then determined. The column with the most ‘1’s is chosen as a possible cut-point. In Table 4.16, the column for cut point ‘A=0.95’ is selected as it is able to discern 6 pairs of objects. After selecting the cut point, those pairs that have been discerned are removed from the partition table. For example, Table

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

4.17 shows the new partition table after selecting ‘A=0.95’ as a cut-point and eliminating the discerned object-pairs. There are only 3 object-pairs left in Table 4.17.

Table 4.16. Partition Table for Cut Points Selection

Object pairs	Possible cut points				
	A=0.5	A=0.75	A=0.95	B=1.45	B=1.85
X ₁ , X ₃	1	1	1	1	1
X ₁ , X ₄	0	1	1	1	0
X ₁ , X ₅	0	0	1	0	0
X ₂ , X ₃	1	1	0	0	0
X ₂ , X ₄	0	1	0	0	0
X ₂ , X ₅	0	0	0	1	1
X ₆ , X ₃	0	0	1	0	1
X ₆ , X ₄	0	1	1	0	0
X ₆ , X ₅	0	0	1	1	0

Table 4.17. Partition Table for Cut Points Selection after First Iteration

Object pairs	Possible cut points				
	A=0.5	A=0.75	A=0.95	B=1.45	B=1.85
X ₂ , X ₃	1	1	0	0	0
X ₂ , X ₄	0	1	0	0	0
X ₂ , X ₅	0	0	0	1	1

The algorithm counting the number of ‘1’s is repeated for the new partition table to select the next cut-point. The second iteration yields the cut point, ‘A=0.75’.

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

Subsequently, the process of getting a new partition table is repeated. The partition table for cut-points selection after the second iteration is shown in Table 4.18.

Table 4.18. Partition Table for Cut Points Selection after Second Iteration

Object pairs	Possible cut points				
	A=0.5	A=0.75	A=0.95	B=1.45	B=1.85
X ₂ , X ₅	0	0	0	1	1

Only 1 object-pair remains to be discerned. There are two cut points that can discern the pair, i.e. at ‘B=1.45’ or at ‘B=1.85’. The cut point was chosen arbitrarily, for example: at ‘B= 1.85’. The point is chosen arbitrarily as choosing any one of them will yield the same number of objects that can be discerned. The algorithm is terminated since all the objects have been discerned. To summarise, the cut points obtained are ‘A=0.75’, ‘A=0.95’ and ‘B=1.85’. Using the cut points, the information system depicted in Table 4.15 can be transformed into a discrete system (Table 4.19).

Table 4.19. The Information System after Discretisation

Objects	Condition Attributes		Decision Attribute
	A	B	D
X ₁	A3	B1	Yes
X ₂	A2	B2	Yes
X ₃	A1	B2	No
X ₄	A1	B1	No
X ₅	A2	B1	No
X ₆	A3	B1	Yes

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

A1, A2, A3, B1 and B2, are used to represent the partitioned spaces, where A1 implies $A \leq 0.75$; A2 indicates $0.75 \leq A \leq 0.95$; A3 represents $A \geq 0.95$; B1 implies $B \leq 1.85$; and B2 indicates $B \geq 1.85$. More meaningful names or symbols such as 'high', 'medium' and 'low' can also be used, if necessary.

4.4.2. Extended MD-Heuristic Algorithm

The MD-Heuristic algorithm described in Section 4.4.1 is applicable to discrete valued decision attributes only. For continuous valued decision attributes, it is not clear how object-pairs with a small difference in values can be discerned. Thus, Step 2 of the MD-Heuristic algorithm cannot be instantiated. To overcome this, an extended MD-Heuristic algorithm is proposed.

Basically, the extension of the MD-Heuristic algorithm aims at facilitating the process of design concept-capability mapping, which is to predict whether a design is able to fulfil the required design capabilities in relation to design specifications. In other words, the mapping mainly concerns whether a set of design concepts is able to meet design specifications. Since design specifications are known prior to the mapping process, this information can be used as the basis to discretise the decision attributes. Accordingly, the partition of the decision attribute space can be used to gauge the ability of an object to meet the stipulated design specifications. For example, a specification demands the total weight (w) of a product to be less than 5 kg. The corresponding decision attribute is partitioned using a single cut point at '5 kg'. Therefore, there are two partitions, ' $0 \leq w \leq 5\text{kg}$ ' and ' $w > 5\text{kg}$ '. Such an approach can be extended to three partitions. As an example, the temperature range (t) of a device is specified to be between 10°C and 50°C . In this case there are two cut points, ' 10°C ' and ' 50°C '. These cut points partition

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

the decision attribute into three partitions, namely ' $t < 10^0\text{C}$ ', ' $10^0\text{C} \leq t \leq 50^0\text{C}$ ' and ' $t > 50^0\text{C}$ ' respectively. This extension is applied to continuous valued decision attributes before applying the MD-Heuristic algorithm. It is a simple extension and yet is able to utilise the essential design specifications.

Table 4.20 shows an information system with 6 objects. Each object has 3 continuous-valued attributes (2 condition attributes and 1 decision attribute). The proposed extension is first applied to partition Decision Attribute D. It is assumed that the specification for Decision Attribute D is ' $1 \leq D \leq 2$ '. Using this information, Decision Attribute D is discretised to yield the information system shown in Table 4.21. In Table 4.21, 'too low' implies ' $D < 1$ ', 'OK' indicates ' $1 \leq D \leq 2$ ', while 'too high' is denoted by ' $D > 2$ '. After Decision Attribute D has been discretised, the aforementioned MD-Heuristic algorithm is used to discretise the condition attributes.

Table 4.20. An Information System with Continuous Valued Attributes

Objects	Condition Attributes		Decision Attribute
	A	B	D
X ₁	1.1	0.9	1.5
X ₂	0.8	1.9	1.7
X ₃	0.4	1.9	0.9
X ₄	0.7	1.8	2.1
X ₅	0.8	0.9	2.8
X ₆	1.1	1.8	1.9

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

Table 4.21. The Information System after Discretisation of the Decision

Objects	Condition Attributes		Decision Attribute
	A	B	D
X ₁	1.1	0.9	OK
X ₂	0.8	1.9	OK
X ₃	0.4	1.9	too low
X ₄	0.7	1.8	too high
X ₅	0.8	0.9	too high
X ₆	1.1	1.8	OK

4.5. Summary

Design concept-capability mapping aims at extracting the relationships between design concepts and design capabilities from past design data. Given a set of design concepts, the mapping, which is represented by a set of design rules, is used to estimate the expected performance. This chapter discusses the various issues in using rough set theory for design concept-capability mapping. It includes multiple decision attributes, non-discrete valued attributes and incomplete information systems. New algorithms to manage these issues have also been proposed.

It has been established that the issue concerning multiple decision attributes can be handled by the proposed attributes decomposition algorithm. Basically, an information system with multiple decision attributes can be split into a number of information systems with single decision attribute. Rules can then be derived from each of these information systems, and subsequently combined using Boolean algebra. It has been shown mathematically that the rules derived using the attributes decomposition

CHAPTER 4 DESIGN CONCEPT-CAPABILITY MAPPING

algorithm are indeed the same as those obtained by considering all the decision attributes simultaneously. Besides simplifying the process for the derivation of rules for multiple decision attributes, it has also been demonstrated that using the proposed algorithm, rules can be easily modified when there is a change in any of the decision attributes. It is apparent that only those rules related to affected decision attributes need to be updated and the need to regenerate the entire rule set has been eliminated.

The issue concerning incomplete information, an algorithm called dissimilar objects algorithm has been proposed. It can be tackled by inserting the value, 'UA', i.e. 'unavailable' to those cells with missing value in an information system. Here, 'UA' is treated as a valid attribute value for the purpose of rule extraction. Once the rules have been derived, those rules with 'UA' values are removed. The treatment of incomplete information is crucial, as it enlarges the search space by allowing the usage of objects with incomplete information in an information system. It has been proven mathematically that the rules derived by the proposed dissimilar objects algorithm are more precise as compared the rules derived without it.

The MD-Heuristic algorithm has been extended to deal with continuous valued condition attributes as well as continuous valued decision attributes. This is made possible via the use of design specifications. The extended MD-Heuristic algorithm has been successfully demonstrated using an example.

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

Chapter 5 Framework of Conceptual Design Assistant

5.1. Introduction

Based on the work described in Chapter 4, a prototype system known as Conceptual Design Assistant (CDA) is proposed here. The relationship of the CDA with the various processes in the conceptual design stage is depicted in Figure 5.1.

Figure 5.1. CDA and Conceptual Design

As shown in Figure 5.1, the CDA performs two main functions, namely estimation of design capabilities (design analysis function) and generation of potential design concepts (design synthesis function) to facilitate the processes in the conceptual design

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

stage. Briefly, the first function of the CDA, estimation of design capabilities, accesses the information stored in a past design database and uses the feedback from designers to generate a set of design rules. The incorporation of the feedback from designers enables the CDA to generate better design rules to be stored in the knowledge base for design concept-capability mapping. Given a set of design specifications, the second function of the CDA, generation of potential design concepts, accesses the knowledge base and the past design database to identify potentially suitable design concepts and introduces them to the designers. As the past design database and the knowledge base are constantly being updated, the CDA is expected to improve its performance over time.

This chapter describes in detail how CDA achieves its analysis and synthesis functions. It also outlines how the performance of the CDA can be improved over time. The details of the design analysis and design synthesis functions are presented in Sections 5.2 and 5.3 respectively. The approach to updating of design rules incrementally is described in Section 5.4. Next, Section 5.5 presents the details of an overall framework of the prototype CDA. The chapter is then summarised in Section 5.6.

5.2. Design Analysis

Using the various approaches described in Chapter 4 for design concept-capability mapping, the design analysis function of the CDA can be realised (Figure 5.2). Basically, the design analysis function embodies a mechanism that is based on the theoretical work described in Sections 4.2 – 4.4, for rule extraction.

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

Figure 5.2. Design Rule Generation

Figure 5.3 shows a step-by-step strategy for the mapping of a set of design concepts into design capabilities. It comprises three modules namely, past design retriever, design concept-capability mapping rule generator and design capability assessor.

Figure 5.3. Design Analysis using the Design Rules Generated

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

Briefly, the past design retriever is a pre-processor that gathers a set of past designs from past design database to form an information system. This information system is used by design concept-capability mapping rule generator to derive a rule set. Subsequently, the set of rules is used by design capability assessor, which is a post-processor, to obtain a set of estimated capabilities. Past design retriever will be elaborated in Section 5.2.1 while design capability assessor will be explored in Section 5.2.2.

5.2.1. Past Design Retriever

The main pre-processing task is to retrieve relevant design data from the past design database for design concept-capability mapping rule generator (Figure 5.3). This past design data need to be treated, as they may be gathered from a wide variety of products. In so doing, reasonable rules can possibly be extracted. The relevant past design data are then used to populate the information system. Subsequently, they are processed by the design concept-capability mapping rule generator. There are numerous approaches available to retrieve relevant past designs. They include the computation approach (Tversky, 1977), the representational approach (Kolodner, 1993) and the hybrid approach that combined the first two approaches (Lenz et. al., 1998). In this work, a past design retrieval mechanism similar to that proposed by Goel et al. (1997) for case based reasoning is adapted. Using such a mechanism, a match between the design purpose of a new design and the past design cases based on certain criteria will trigger the retrieval process.

More formally, using the notation of the PBS representation presented in Section 3.3.4, the purpose domain of a new design object (P_n) can be expressed as follows.

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

$$P_n = \{ P_{nE}, P_{nA}, P_{nV}, P_{nR} \}$$

where

P_{nE} = a finite set of purpose element nodes,

P_{nA} = a finite set of purpose attribute nodes,

P_{nV} = a finite set of purpose value nodes, and

P_{nR} = a finite set of purpose relation edges.

Accordingly, the purpose domain of a past design object (P_p) is given by

$$P_p = \{ P_{pE}, P_{pA}, P_{pV}, P_{pR} \}$$

where

P_{pE} = a finite set of purpose element nodes,

P_{pA} = a finite set of purpose attribute nodes,

P_{pV} = a finite set of purpose value nodes, and

P_{pR} = a finite set of purpose relation edges.

Thus, the similarity measure between a new design and a past design can be expressed using Definition 5.1 as follows.

$$\phi = \frac{\text{card}(P_{nE} \cap P_{pE})}{\text{card}(P_{nE})} \quad \text{Definition 5.1}$$

where

$\text{card}(P_{nE} \cap P_{pE})$ = the number of elements in the set, $P_{nE} \cap P_{pE}$, and

$\text{card}(P_{nE})$ = the number of elements in the set, P_{nE} .

A threshold level $\bar{\phi}$ can be set to determine whether a past design can be included in the information system. If $\phi \geq \bar{\phi}$, the past design is considered sufficiently similar to the

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

new design, as it contains a sufficient number of design purposes that are applicable to the new design. The value of $\bar{\phi}$ is set to '1' if only those past design objects with identical design purpose are to be considered. On the other hand, $\bar{\phi}$ is set to '0' if all past design objects are to be included in the analysis.

Other than extracting useful past design data, it is necessary to select those attributes that are relevant to design rule generation. Note that not all the condition attributes of an information system have an impact on design decisions, i.e. decision attributes. The omission of an important condition attribute or the inclusion of an unrelated condition attribute would result in superfluous design rules being extracted. However, if the information system is sufficiently comprehensive, which may not always be the case in reality, the approaches developed in this work is able to sift out unrelated attributes automatically.

To illustrate, consider a four-attribute information system for the design of an electric kettle (Table 5.1). From the design rules generated (Table 5.2), Condition Attribute Colour seems closely related to Decision Attribute Temperature. Intuitively, such a deduction is not true, as colour is unrelated to the determination of water temperature for an electric kettle.

There are two methods to overcome such an erroneous deduction:

- a) Use prior knowledge to remove unrelated attributes, if any, from an information system before rule generation, or
- b) Include more objects on the information system.

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

Table 5.1. An Information System with an Unrelated Condition Attribute, ‘Colour’

Objects	Condition Attributes			Decision Attribute
	Voltage	Current	Colour	Temperature
X ₁	low	low	red	low
X ₂	low	high	red	low
X ₃	high	low	yellow	medium
X ₄	high	high	yellow	high

Table 5.2. Rule set Derived from Table 5.1

NO	IF	THEN
1	((Volt=low)+(Col=red))	Temp=low
2	((Volt=low)+(Col=red))	Temp=low
3	((Volt=high)+(Col=yellow))(Cur=low)	Temp=med
4	((Volt=high)+(Col=blue))(Cur=high)	Temp=high

Note: Volt: Voltage; Col: Colour; Cur: Current; Temp: Temperature

Method (a) can be easily implemented via user input. As for Method (b), it can be illustrated by appending four additional objects to Table 5.1 (Table 5.3). The resulting rules are depicted in Table 5.4.

The inclusion of 4 additional objects, Objects X₅ to X₈, has effectively removed Condition Attribute Colour from the design rule set. This implies that if the past design database is sufficiently comprehensive, more reasonable design rules can possibly be extracted.

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

Table 5.3. A More Comprehensive Information System

Objects	Condition Attributes			Decision Attribute
	Voltage	Current	Colour	Temperature
X ₁	low	low	red	low
X ₂	low	high	red	low
X ₃	high	low	yellow	medium
X ₄	high	high	yellow	high
X ₅	low	low	yellow	low
X ₆	low	high	yellow	low
X ₇	high	low	red	medium
X ₈	high	high	red	high

Table 5.4. Rule set Derived from Table 5.3

NO	IF	THEN
1	(Volt=low)	Temp=low
2	(Volt=low)	Temp=low
3	(Volt=high)(Cur=low)	Temp=med
4	(Volt=high)(Cur=high)	Temp=high
5	(Volt=low)	Temp=low
6	(Volt=low)	Temp=low
7	(Volt=high)(Cur=low)	Temp=med
8	(Volt=high)(Cur=high)	Temp=high

Note: Volt: Voltage; Col: Colour; Cur: Current; Temp: Temperature

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

While the inclusion of additional design objects can help in removing unrelated attributes, these objects may not be available in reality. Thus, it is more reasonable to adopt Method (a) to eliminate unrelated attributes for practical reasons. Accordingly, if a condition attribute is not associated with any decision attribute, it is immediately omitted before rule extraction.

It is important to highlight that unrelated attributes as described in this section is not the same as incomplete information discussed in Section 4.3. Unrelated attributes refer to those condition attributes that have no impact whatsoever on the decision attributes that are being predicted. On the other hand, incomplete information as described in Section 4.3 concerns the usage of some objects that contains incomplete information due to some of the condition attributes being non-relevant to those objects. These condition attributes themselves are actually related to the decision attributes that are being predicted.

5.2.2. Design Capability Assessor

After the information system has been populated, the design concept-capability mapping rule generator module is invoked to yield a set of design rules. The design rules obtained can then be used to estimate the capabilities of a new design. Specifically, the specifications of a new design are first transformed into attribute-value pairs. These attribute-value pairs are then evaluated using the above mentioned design rules to produce a set of corresponding capabilities, i.e. design concept-capability mapping. At times, the evaluation may result in more than one estimation for a capability of a new design due to the triggering of multiple design rules. A reliability measurement scheme to facilitate the selection of appropriate capabilities is therefore

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

needed and proposed. In this work, a simple tally of the total number of rules that trigger a particular capability is used as the reliability measurement.

More formally, using Definition 4.5, it is possible to define a set of rules, R_N , which is triggered by the attribute-value pairs, i.e. design specifications, of a new design, X_N as follows.

$$R_N = \{r_{[C,D]} : \bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C,D]}(X_i, X_j) = C(X_N)\} \quad \text{Definition 5.2}$$

where $C(X_N)$ is the value of Condition Attribute C of Object X_N .

Accordingly, the total number of rules is given by $\text{card}(R_N)$. In other words, this set of rules yields $\text{card}(R_N)$ number of possible estimated capabilities, which may or may not be identical. Let these capabilities be grouped within a set, $D(X_N)$.

$$D(X_N) = \{D(X_N)_i : \bigcap_{j \in \{1, \dots, n\}} \bigcup \tilde{m}_{[C,D]}(X_i, X_j) = C(X_N)\} \quad \text{Definition 5.3}$$

Since the estimated capabilities in Set $D(X_N)$ may or may not be identical, Set $D(X_N)$ can be partitioned into k number of subsets $D(X_N)^k$ where k is the number of unique capabilities

$$D(X_N)^k = \{D(X_N)_i : D(X_N)_i = D(X_N)_j\} \quad \text{Definition 5.4}$$

Thus, the reliability measure of capabilities, η^k , is given by

$$\eta^k = \frac{\text{card}(D(X_N)^k)}{\text{card}(D(X_N))} \quad \text{Definition 5.5}$$

It can be deduced that the range of η^k is between 0 and 1 ($0 < \eta^k \leq 1$). Also, it is obvious that when $\eta^k = 1$, the estimated capability is unique.

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

5.3. Design Synthesis

The concept of design concept-capability mapping described in Chapter 4 can be extended to design synthesis. As previously mentioned in Section 5.2, the purpose of performing design concept-capability mapping is to generate a set of rules that maps the attribute-value pairs of the design concept domain into attribute-value pairs of the design capability domain. Essentially, this is a single-direction mapping process, i.e. design concept to design capability, which links the two domains, namely design concepts domain and capabilities domain, in product design.

Conversely, it is postulated that the mapping process can be bi-directional, i.e. design concepts domain to capabilities domain or vice versa. By enabling the mapping of capabilities domain to design concepts domain, it is possible to carry out design synthesis, where design capabilities and design concepts are used as condition attributes and decision attributes respectively. Figure 5.4 shows the process of design synthesis, which is fairly similar to that of design analysis (Figure 5.3).

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

Figure 5.4. Design Synthesis using Design Concept-Capability Mapping

Basically, in design synthesis, designers begin with design specifications, i.e. a subset of design capabilities, and aim at generating a set of design concepts. Thus, a set of past designs can be retrieved using the same method described in Section 5.2.1.

To recapitulate, the past design data are retrieved based on the similarity measure of their respective design purposes. They are then utilised to generate a set of design rules for design capability-concept mapping. Using the design specifications provided, the set of rules is employed to derive design concepts that can possibly satisfy these specifications. Similar to design analysis, more than one design concepts can be obtained using design synthesis. This implies that the measure of reliability, η^k , can also be used in design synthesis to narrow down the choice.

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

5.4. Incremental Updating of Rules

Rules derived using rough set theory inherently need to be updated whenever a new object is added to the information system. From the literature review described in Section 2.6.5, it has been established that existing approaches to the updating of rules mainly concern regeneration of design rule set using the existing information system appended with these new objects (Shan and Ziarko, 1995). Tong and An (2002) extended Shan and Ziarko's approach by including additional algorithms to determine the coverage, confidence level and support of the rules generated. In general, the extraction of rules from an information system appended with new objects or additional past design data can be achieved by the aforementioned approaches. This, however, requires the entire information system to be revisited. Attempts to simplify the rule extraction process via incremental updating of rules, thus far, have been lacking. This section describes the work that explores the possibility of realising incremental updating of design rules for design concept-capability mapping.

5.4.1. Approaches to Updating of Design Rule Base

5.4.1.1. Ad-hoc Updating Approach

The ad-hoc updating approach offers the simplest form of design rules updating (Figure 5.5). It aims to generate a rule set to be used when necessary. It comprises 3 basic steps, namely past design retrieval, discernibility matrix construction and rule generation. Given a new design task that consists of a set of attributes and values, the approach will perform a thorough search in the past design database for similar past design objects. This is done by comparing the attributes of the current design task with those in the past

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

design database. Utilising the similarity measure described in Section 5.2.1, a set of past design objects can be retrieved. The discernibility matrix corresponds to this set of past design objects is then constructed. Subsequently, a complete set of design rules associated with this discernibility matrix can be obtained by simplifying the entries of the matrix.

Figure 5.5 Ad-hoc Updating Approach

As the design rule set is linked with the discernibility matrix as well as the set of past design objects that is used initially, any changes to the past design objects used is likely to render the rule set invalid. Thus, when a new ‘past design’ has been added to the past design database or an existing design in the past design database has been modified, the above-mentioned procedure has to be repeated in order to ensure the validity of the rule set. It can be inferred that as the past design database grows larger, the rule extraction process is likely to become more cumbersome, as it is necessary to search through the entire past design database in order to retrieve all the relevant past designs. Thus, an approach to facilitate updating of design rules is needed.

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

5.4.1.2. Direct Updating Approach

The direct updating approach aims at ensuring the validity of the design rule set as the past design database expands. It attempts to improve the efficiency of design rule updating by exploring the possibility of re-using the rules generated previously. The proposed approach is depicted in Figure 5.6.

Figure 5.6 Direct Updating Approach

Compared to the ad-hoc updating approach described in Section 5.4.1.1, a design rule database that is used to store the rules generated has been incorporated. This design rule database needs to be updated regularly. This updating of design rule database is done through a batch process, which is indicated by the dotted lines shown in Figure 5.6. Basically, the newly appended past designs are checked against the design rules stored in the design rule database. If they are considered to be similar (based on the similarity measure described in Section 5.2.1) with the past design objects used to generate a particular rule set, the newly appended past designs are retrieved. Subsequently, the design rule database is updated by:

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

1. Assembling a discernibility matrix containing all the relevant attribute information,
2. Simplifying the entries of the discernibility matrix using the approach described in Section 4.2.2, and
3. Extracting the design rule set and updating the design rule database.

Compared to the ad-hoc rules updating approach described in Section 5.4.1.1, there is no need to search through the entire past design database in order to extract past designs with relevant attributes for updating the design rule database. Using the direct updating approach, only newly appended past designs and relevant past are retrieved. For the purpose of design analysis or design synthesis, a relevant rule set is retrieved from the design rule database. The retrieval is triggered by comparing the set of attributes contained in the rule set with those of the new design task.

However, changes in design task during design iteration may yield a different set of design attributes. Accordingly, the design rule sets generated earlier may become irrelevant to the current design task. As design changes are expected to be carried out fairly frequently during conceptual design, multiple sets of design rules need to be generated and stored in the design rule database. It is likely that many of these rule sets may not even be re-used in design analysis or design synthesis due to mismatch of design attributes. This results in wastage of computing resources and effort to update the design rule database. Thus, it is anticipated that a more efficient approach to updating of design rule sets is necessary.

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

5.4.1.3. The Incremental Updating Approach

The incremental updating approach attempts to alleviate the problems of generating multiple design rule sets. A so-called global discernibility matrix is proposed and implemented in this work. Essentially, this matrix provides a depository to register all the discernibility matrices that are used to generate the design rule sets as a design evolves over time.

To accommodate the CDA functions, there are two global discernibility matrices, one for design analysis and the other for design synthesis. In design analysis, the condition attributes and decision attributes are design concepts and design capabilities respectively. As for design synthesis, the roles of design concepts and design capabilities are reversed.

A global discernibility matrix, $\overline{\overline{M}}$, basically records the attributes that discern pairs of objects in the past design database. Mathematically, it can be represented by Definition 5.6 as follows. Given a population of past design objects consisting of n objects and m decision attributes,

$$\overline{\overline{M}}_{[C,D]} = \{\overline{\overline{M}}_{[C,dk]} : 1 \leq k \leq m\} \quad \text{Definition 5.6}$$

where $\overline{\overline{M}}_{[C,dk]}$ is defined as:

$$\overline{\overline{M}}_{[C,dk]} = \{m_{[C,dk]}(X_i, X_j) : 1 \leq i \leq n, 1 \leq j \leq n\}$$

where $m_{[C,dk]}$ is defined as:

for $1 < i < n; 1 < j < n$

$$m_{[C,dk]}(X_i, X_j) = \begin{cases} m_{[C]}(X_i, X_j) & \text{if } dk(X_i) \neq dk(X_j) \\ 1 & \text{otherwise} \end{cases}$$

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

where $m_{[C]}(X_i, X_j)$ is defined as:

$$m_{[C]}(X_i, X_j) = \{a \in C : a(X_i) \neq a(X_j)\} \text{ for } 1 < i < n; 1 < j < n$$

Definition 5.6 can be represented graphically by Figure 5.7. As shown, there are m layers in the global discernibility matrix. In each layer, there is an $n \times n$ table. Each of the $n \times n$ table records the condition attributes that discern pairs of objects for a single decision attribute. In other words, it denotes a relative discernibility matrix in relation to the corresponding decision attribute. Each of the cells of the relative discernibility matrix in each layer represents the attributes that discern a pair of objects, i.e. Objects X_1 - X_2 , X_1 - X_3 , and so on.

Figure 5.7. Layers of A Global Discernibility Matrix

Accordingly, any part of the global discernibility matrix can be retrieved depending on the design task that needs to be carried out. In other words, relevant layers of the matrices are identified based on the decision attributes. For the global discernibility

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

matrix that is intended for design analysis, these layers correspond to the intended design capabilities to be predicted. As for the matrix intended for design synthesis, the layers correspond to design concepts.

The flow of the approach for incremental updating of rules is shown in Figure 5.8. Here, the similarity measure (Section 5.2.1) is used to identify those past design data that are relevant to the new design task. Upon completion, the information stored in the cells corresponding to these past designs is retrieved. This information is then employed to form a temporary discernibility matrix. Subsequently, it is used to produce a design rule set that can be used for design analysis or design synthesis.

Figure 5.8 Approach for Incremental Updating of Rules

To maintain the relevance of the two global discernibility matrices for design analysis and design synthesis, design attributes of new or modified designs are compared with other past designs. The comparison results obtained comprising attributes that discern object pairs are then appended to the global discernibility matrix. In so doing, the global discernibility matrix is updated incrementally.

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

5.4.1.4. Comparison of the Various Approaches to Updating of Rules

The performance of the three approaches to the updating of design rules is analysed qualitatively. Two considerations regarding updating speed and operational speed need to be taken into account. As updating speed could be done in the background, it needs not be addressed here. As for operational speed, it can be argued that the ad-hoc updating approach is the worst performer. Its operational speed is inherently slow as all the processes required for rule generation are required to be carried out when updating of rules is taking place. Basically, ad-hoc updating approach needs to search thoroughly through the past design database every time when it is invoked. As for direct updating and incremental updating, at the first glance, the direct updating approach appears to have a better operational speed as compared to that of the incremental updating approach, as it requires fewer processes to be carried out. However, such an observation is based on the assumption that the set of design rules is available in the design rule database. In the event where the set of design rules has not been generated, the operational speed of the direct updating approach is slower since it has to carry out all the processes similar to those of the ad-hoc updating approach. This is where the incremental updating approach, which is done via updating the global discernibility matrix, is expected to outperform the direct updating approach.

Arguably, both the direct updating approach and the incremental updating approach have their own advantages and disadvantages. However, as conceptual design involves design revisions, it is unlikely that a set of matching design rules is always made available. The reason being reliable design rules are problem specific, i.e. their condition attributes and decision attributes need to match the design problem at hand.

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

Thus, the incremental updating approach is more appropriate for the proposed design-concept capability mapping.

5.4.2. Details of the Incremental Updating Approach

5.4.2.1. The Algorithm

The algorithm for the implementation of the proposed incremental updating approach is summarised as follows.

Updating of the global discernibility matrices

1. Retrieve newly added or modified past design.
2. Compare the attribute values of the new or modified past design with those stored in the past design database to generate attributes that discern object-pairs.
3. Modify the cells in global discernibility matrices related to the new or modified past design.

Usage of the global discernibility matrices

1. Identify relevant layers in the global discernibility matrices according to the decision attributes.
2. Retrieve relevant cells on the layers identified based on the preset threshold values to form a temporary discernibility matrix.
3. Simplify the entries of the cells of the temporary discernibility matrix.
4. Generate and simplify the design rules.

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

5.4.2.2. Illustration

To illustrate, the example presented in Section 4.2 was revisited. An extract of the past design database is shown in Table 5.5. Here, only the global discernibility matrix for design analysis will be discussed as the global discernibility matrix for design analysis is very similar in principle. It was assumed that the global discernibility matrix did not have any entry initially.

Table 5.5. An Extract of the Past Design Database

Last Update	Design Objects	Condition Attributes					Decision Attributes	
		A	B	C	D	E	F	G
1-Jan-05	X ₁	1	1	1			1	1
2-Jan-05	X ₂	1	1	2			2	1
3-Jan-05	X ₃	2	2	1			3	1
4-Jan-05	X ₄	2	1	2			1	1
5-Jan-05	X ₅	3	1	1			2	2
5-Jan-05	X ₆	3	1	1			3	2
6-Jan-05	X ₇	1			1	2		2
6-Jan-05	X ₈		2		1	3		2
6-Jan-05	X ₉				1	2	1	

Assume that there is a need to generate a set of rules for design analysis of a new design on 7 January 2005. It is further assumed that the design involves Condition Attributes A, B and C, and Decision Attributes F and G. A similarity threshold value of 75% was used. Design Objects X₇, X₈ and X₉ were omitted as they only have similarity measure of 57%, 57% and 43% respectively. Thus, the relative discernibility matrices for the

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

generation of design rules could be obtained (Tables 5.6 – 5.7). Based on these matrices, the complete set of rules was generated (Table 5.8).

Table 5.6. Relative Discernibility Matrix for Decision Attribute F

7-Jan-05	X ₁	X ₂	X ₃	X ₄	X ₅	X ₆
X ₁	1	C	AB	1	A	A
X ₂	C	1	ABC	A	1	AC
X ₃	AB	ABC	1	BC	AB	1
X ₄	1	A	BC	1	AC	AC
X ₅	A	1	AB	AC	1	1
X ₆	A	AC	1	AC	1	1

Table 5.7. Relative Discernibility Matrices for Decision Attribute G

7-Jan-05	X ₁	X ₂	X ₃	X ₄	X ₅	X ₆
X ₁	1	1	1	1	A	A
X ₂	1	1	1	1	AC	AC
X ₃	1	1	1	1	AB	AB
X ₄	1	1	1	1	AC	AC
X ₅	A	AC	AB	AC	1	1
X ₆	A	AC	AB	AC	1	1

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

Table 5.8. Rule Set for Decision Attributes F and G

7-Jan-05	IF	THEN
1	(A=1)(C=1)	(F=1)(G=1)
2	(A=1)(C=2)	(F=2)(G=1)
3	((A=2)+(B=2))((B=2)+(C=1))	(F=3)(G=1)
4	(A=2)((B=1)+(C=2))	(F=1)(G=1)
5	(A=3)	(F=2)(G=2)
6	(A=3)	(F=3)(G=2)

These relative discernibility matrices were then stored in the global discernibility matrix for design analysis. Subsequently, on 8 January 2005, Design Object X_3 was modified and Design Object X_{10} had been appended to the past design database. On 9 January 2005, the incremental updating approach was invoked.

A new design object, i.e. Design Object X_{10} , as well as a modified design object, i.e. Object X_3 , had been identified. Accordingly, the past design database was modified to accommodate these changes. A partial information system presented in Table 5.9 is used to illustrate the changes.

Table 5.9. A Modified Information System (Partial)

Last Update	Design Objects	Condition Attributes			Decision Attributes	
		A	B	C	F	G
8-Jan-05	X_3	4	2	1	4	1
8-Jan-05	X_{10}	2	2	2	4	2

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

These changes would have an impact on the global discernibility matrix generated on 7 January 2005. Thus, the global discernibility matrix needed to be updated. This was achieved by updating the global discernibility matrix using the information given in Table 5.9. The changes are shown in Tables 5.10 and 5.11.

Table 5.10. Updated Layer of the Global Discernibility Matrix for Decision Attribute D

	X ₁	X ₂	X ₃	X ₄	X ₅	X ₆	X ₇	X ₈	X ₉	X ₁₀
X ₁	1	C	AB	1	A	A				ABC
X ₂	C	1	ABC	A	1	AC				AB
X ₃	AB	ABC	1	ABC	AB	AB				1
X ₄	1	A	ABC	1	AC	AC				B
X ₅	A	1	AB	AC	1	1				ABC
X ₆	A	AC	AB	AC	1	1				ABC
X ₇										
X ₈										
X ₉										
X ₁₀	ABC	AB	1	B	ABC	ABC				1

Assume that a design analysis was to be carried out on 10 January 2005. This required a rule set to be generated. Accordingly, the entries for Design Objects X₁ to X₆ and X₁₀ were retrieved to form the temporary relative discernibility matrices (Tables 5.12 and 5.13). Table 5.14 shows the complete rule set generated.

These two layers of the global discernibility matrix were arranged in a 3-D like matrix as explained in Section 5.4.1.3. Only the shaded cells in the tables (Tables 5.10 and 5.11) were modified using the incremental updating approach. The entries of the cells

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

corresponding to Design Objects X₇, X₈ and X₉ were blank. Hence, they were not used in the generation of the temporary discernibility matrices.

Table 5.11. Updated Layer of the Global Discernibility Matrix for Decision Attribute E

	X ₁	X ₂	X ₃	X ₄	X ₅	X ₆	X ₇	X ₈	X ₉	X ₁₀
X ₁	1	1	1	1	A	A				ABC
X ₂	1	1	1	1	AC	AC				AB
X ₃	1	1	1	1	AB	AB				AC
X ₄	1	1	1	1	AC	AC				B
X ₅	A	AC	AB	AC	1	1				1
X ₆	A	AC	AB	AC	1	1				1
X ₇										
X ₈										
X ₉										
X ₁₀	ABC	AB	AC	B	1	1				1

Table 5.12. Temporary Relative Discernibility Matrix for Decision Attribute D

	X ₁	X ₂	X ₃	X ₄	X ₅	X ₆	X ₁₀
X ₁	1	C	AB	1	A	A	ABC
X ₂	C	1	ABC	A	1	AC	AB
X ₃	AB	ABC	1	ABC	AB	AB	1
X ₄	1	A	ABC	1	AC	AC	B
X ₅	A	1	AB	AC	1	1	ABC
X ₆	A	AC	AB	AC	1	1	ABC
X ₁₀	ABC	AB	1	B	ABC	ABC	1

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

Table 5.13. Temporary Relative Discernibility Matrix for Decision Attribute E

	X ₁	X ₂	X ₃	X ₄	X ₅	X ₆	X ₁₀
X ₁	1	1	1	1	A	A	ABC
X ₂	1	1	1	1	AC	AC	AB
X ₃	1	1	1	1	AB	AB	AC
X ₄	1	1	1	1	AC	AC	B
X ₅	A	AC	AB	AC	1	1	1
X ₆	A	AC	AB	AC	1	1	1
X ₁₀	ABC	AB	AC	B	1	1	1

Table 5.14. Updated Complete Rules based on Decision Attributes D AND E

10-Jan-05	IF	THEN
1	(A=1)(C=1)	(D=1)(E=1)
2	(A=1)(C=2)	(D=2)(E=1)
3	((A=4)+(B=2))((A=4)+(C=1))	(D=4)(E=1)
4	(A=2)(B=1)	(D=1)(E=1)
5	((A=3)+(B=1))((A=3)+(C=1))	(D=2)(E=2)
6	((A=3)+(B=1))((A=3)+(C=1))	(D=3)(E=2)
7	(B=2)((A=2)+(C=2))	(D=4)(E=2)

From Table 5.14, it was apparent that the rule set contained 7 rules. As expected, Rule 3 had been altered. The alteration was due to the change in design attributes for Design Object X₃. As for Rules 1 and 2, no change was observed. Although the changes in the global discernibility matrices due to the new Design Object X₁₀ and a modified Design

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

Object X_3 were quite minimum, viz. only the entries of some of the cells had been modified, the impact created on the rule set was quite evidence. For example, Rule 4 of Table 5.15 was quite different compared to that of Table 5.9.

At the first glance, it could be argued that Rule 4 should not change, as the attributes of the corresponding design object, X_4 , had not been modified. Analysis shows that the change in Rule 4 was effected by the changes in Design Object X_3 . In other words, the design rules generated by the incremental updating approach could be easily traced. Further analysis was carried out to ascertain the validity of the rules generated by the incremental updating approach. Based on the same assumptions, design rules were generated using the ad-hoc updating approach. It was found that identical set of discernibility matrices, and hence the same rule set, could be obtained.

5.4.2.3. Discussion on the Approach for Incremental Updating of Rules

It is a well-known fact that design evolves over time. As new design objects or more past design data are made available, they are appended onto the past design database, and subsequently influence the design rules generated. As mentioned previously, using the approach proposed by Shan and Ziarko (1995), an updated set of rules can be obtained. The ad-hoc updating approach basically applies the approach by Shan and Ziarko as and when needed. The incremental updating approach requires the information registered in the so-called global discernibility matrices to be updated. Essentially, the global discernibility matrices are depositories of past design knowledge, which are obtained from past design database. By updating the global discernibility matrices incrementally, the design knowledge registered can be improved progressively.

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

The incremental updating approach simplifies the process of rule extraction. Using such an approach, rule extraction is driven by needs. More specifically, instead of analysing the entire past design database, a subset of the global discernibility matrices, which is selected based on the intended design attributes, is used. When incorporated as part of the proposed CDA, it is envisaged that the resulting system is able to respond to different requirements dynamically. Such a property is well suit for design, as design, conceptual design in particular, evolves over time.

5.5. A Framework for a Prototype Conceptual Design Assistant

A framework for the development of the proposed conceptual design assistant (CDA) is shown in Figure 5.9. It comprises 6 modules, namely a design concept-capability mapping mechanism, a past design retriever, an incremental updating mechanism, a design concept extractor, a design capability assessor and a graphical user interface for the input of design specifications and design purposes from designers.

5.5.1. Design Concept-Capability Mapping Mechanism

Briefly, the design concept-capability mapping mechanism forms a core module of the prototype system. It is linked to the past design retriever, the incremental updater, the design concept extractor, the design capability assessor and the graphical user interface (GUI). The GUI enables the designers to provide inputs to the CDA. As shown in Figure 5.9, the design concept-capability mapping mechanism is used to solicit information through the GUI, update and retrieve information from the past design database and the global discernibility matrices, instantiate the various modules, coordinate the inputs provided by the stakeholders to the modules, and provide advice

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

for conceptual design. As the core module of the prototype system, it also acts as a design rule generator, which is based on the approaches described in Chapter 4. Using an information system derived by the past design retriever, the design concept-capability mapping mechanism proceeds to produce design rules for subsequent deployment.

Figure 5.9. An Overall Framework for CDA

5.5.2. Past Design Retriever

The past design retriever provides the inputs in the form of an information system to the design-concept capability mapping mechanism for the derivation of design rules. The approach described in Section 5.2.1 is implemented in this module. The information system comprises a collection of condition attributes and decision attributes. As

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

mentioned previously, in design analysis, the condition attributes and the decision attributes are associated with design concepts and design capabilities respectively. As for design synthesis, the relationships are reversed.

5.5.3. The GUI

The GUI provides a communication gateway between the CDA and the stakeholders. It enables the stakeholders to enter design specifications and design purposes for design rule generation. The results obtained from design analysis and design synthesis are presented to the users through the GUI. The GUI also provides an interface that enables the users to remove unrelated attributes as mentioned in Section 5.2.1.

5.5.4. Design Capability Assessor

As already mentioned in Section 5.1, predicting design capabilities, i.e. design analysis, constitutes one of the main functions of the CDA. Basically, the design rules generated by the design concept-capability mapping mechanism are used to assess the design capabilities and presented them to designers. Hence, the reliability of the design rules generated by the mechanism needs to be ascertained before using them to predict design capabilities. As the mapping of design capabilities may yield more than one estimations, the reliability measure mentioned in Section 5.2.2 has also been incorporated into the prototype system. This enables appropriate design capability to be selected.

Through the design concept-capability mapping mechanism, design capability assessor is linked with the GUI. This linkage enables the designers to carry out design analysis or design synthesis. The interaction between the designers and the CDA is shown in Figure 5.10.

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

Figure 5.10. Design Capability Assessor and User Interactions

The designers generate a set of design concepts, which are evaluated by the prototype CDA. Using the design concept-capability mapping mechanism, the design capabilities of these design concepts can be estimated. With the reliability measure determined, the estimated design capabilities are forwarded to the GUI for evaluation. Through the GUI, the designers have the options to:

- Refine the design concepts and/or design specifications if the assessment suggests that the design concepts obtained are unlikely to meet design specifications, or
- Ignore the assessment if it appears to be erroneous.

If the design concepts and/or design specifications are refined, further assessment of the design concepts needs to be carried out.

5.5.5. Design Concept Extractor

This module performs the second function of the CDA, i.e. design synthesis. Based on the mapping provided by the design concept-capability mapping mechanism, potential

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

design concepts can be presented to the designers via the GUI. The designers can then judge the merits of these design concepts and, if needed, derive new design concepts. The function of the design concept extractor is shown in Figure 5.11.

Figure 5.11. Design Concept Extractor and User Interactions

Figures 5.10 and 5.11 are fairly similar except the input and output of the CDA. As already mentioned, CDA comprises two functions, namely design analysis and design synthesis. The input for design analysis is the designer action to generate design concepts while the input for design synthesis is the designer action to generate design purpose domain. As for the outputs, in design analysis designers can refine design specifications and/or design concepts, while in design synthesis designers can refine design specifications and/or design purposes.

5.5.6. Incremental Updater

Similar to expert designers, it is highly desirable if the CDA is able to accumulate or acquire design knowledge progressively. To realise this, an incremental updater has

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

been incorporated into the overall framework. The incremental updater adopts the incremental updating approach presented in Section 5.4.2.1. It is invoked regularly to maintain the global discernibility matrices.

5.6. Operational Interactions of the Modules

The main functions of the modules are depicted in Table 5.15. Figure 5.12 shows the functions of the prototype system as well as the associated databases.

Table 5.15. Functionalities of the Modules

Modules	Functions
Design Concept-Capability Mapping Mechanism	Control flow
	Generate rules
Past Design Retriever	Search for past designs
	Setup information systems
GUI	Receive design concepts
	Receive design specifications
	Receive feedbacks
Design Capability Assessor	Predict design capabilities
	Present estimated capabilities
Design Concept Extractor	Search for design concepts
	Present design concepts
Incremental Updater	Search for updated past designs
	Update global discernibility matrices

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

Note: DCCM Mechanism = Design Concept-Capability Mapping Mechanism

Figure 5.12. Interactions between Modules

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

The operational interactions of the various modules are summarised as follows. Design capability assessor is invoked when a new design is to be evaluated. In this situation, the inputs to the CDA are design concepts, which are received through the GUI. Design concept extractor is activated when a designer requests for the extraction of possible design concepts. The inputs to the design concept extractor are design specifications that are entered through the GUI.

After the results (design capabilities for design analysis and design concepts for design synthesis) have been presented to the designers, their feedbacks can be solicited via the GUI. These feedbacks can be used by the design concept-capability mapping mechanism to improve the mapping by modifying design rules.

As for the incremental updater, it checks the global discernibility matrices to determine if any cells of the global discernibility matrices need to be updated by the approach presented in Section 5.4.2. This is to be done regularly and in the background when the prototype system is not in use.

5.7. Summary

This chapter describes an overall framework of a prototype Conceptual Design Assistant (CDA). The CDA comprises 6 modules, namely design concept-capability mapping mechanism, past design retriever, GUI – inputs for specifications and feedback, design capability assessor, design concept extractor, and incremental updater. Briefly, the design concept-capability mapping mechanism is a coordinator, which provides the control and the mapping facilities for the prototype CDA. Past design retriever is responsible for extracting relevant past design data that are organised

CHAPTER 5 FRAMEWORK OF CONCEPTUAL DESIGN ASSISTANT

according to the PBS representation scheme described in Chapter 3. These past design data are identified using the similarity measure developed in this work. Subsequently, they are formulated as an information system for rule extraction using the various approaches presented in Chapter 4. The GUI offers an interface for the prototype CDA to communicate with the designers and stakeholders. Through the GUI, inputs and feedbacks can be solicited. Design capability assessor and design extractor embody the design analysis function and design synthesis function respectively. It is a well-known fact that the design of a product evolves over time. The evolution may bring about changes in past design database and hence the design rules. Therefore, there is a need to look into ways to update design rules incrementally. A detailed analysis on the various ways in which incremental updating of rules can be achieved has been performed. A novel knowledge representation scheme has also been proposed to facilitate the updating process. Such a knowledge representation scheme uses a so-called global discernibility matrix to record those attributes that are able to discern object-pairs on an information system. As the CDA is tasked to perform two main functions, i.e. design analysis and design synthesis, two global discernibility matrices are therefore needed. In this work, incremental updating of rules is achieved by updating the global discernibility matrices, extracting relevant information from the resulting matrices to form temporary discernibility matrices, and carrying out rule extraction from the temporary matrices using the various approaches presented in Chapter 4. In so doing, design rules can be updated incrementally. In the proposed CDA, the incremental updater is developed to embed the approach. The implementation of the incremental updater enables the CDA to respond to different requirements dynamically.

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

Chapter 6 System Implementation and Case Studies

6.1. System Implementation

6.1.1. Overall System

Figure 6.1. System Overview: User Interfaces, Modules and Databases

The implementation of a framework of the prototype CDA described in Chapter 5 is shown in Figure 6.1. As shown, it comprises 4 user interfaces, 9 modules and 2 databases. Their functionalities are summarised in Table 6.1. The CDA performs 4 main

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

functions, namely updating of past designs, updating of global discernibility matrices, design analysis and design synthesis designs.

Table 6.1. System Functions and Interface/Module Description

Function	Interface/Module	Descriptions
A: Updating of Past Design	Past Design Update Interface	An interface for a designer to work on a current design or to modify a past design
	Past Design Update Module	A module to retrieve and store a design (current or past)
B: Updating of Global Discernibility Matrices	Global Discernibility Matrix Update Interface	An simple interface for a designer to initiate an update of the global discernibility matrix
	Global Discernibility Matrix Checking Module	A module to check for new changes in the past design database and compare it to last update in entries in discernibility matrix
	Global Discernibility Matrix Update Module	A module to update the update columns and rows in the matrix that needs updating
C: Design Analysis	Concept Input Interface	An interface for a designer to input design concepts need to be analysed
	Temporary Discernibility Matrix Retrieval Module	A module to retrieve relevant rows and columns from discernibility matrix database to form a temporary discernibility matrix
	Rule Construction Module	A module to transform the temporary discernibility matrix into a set of rules
	Rule Trigger Module	A module that uses the set of rules generated to perform estimation for design capabilities
	Capability Display Interface	An interface to display the estimated capabilities to the designer
D: Design Synthesis	Specification Input Interface	An interface for a designer to input design specifications to derive design concepts
	Temporary Discernibility Matrix Retrieval Module	A module to retrieve relevant rows and columns from discernibility matrix database to form a temporary discernibility matrix
	Rule Construction Module	A module to transform the temporary discernibility matrix into a set of rules
	Rule Trigger Module	A module that uses the set of rules generated to perform estimation for design capabilities
	Concept Display Interface	An interface to display the estimated design concepts to the designer

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

The system was implemented using a personal computer with Windows 2000 operating system. Visual C++ was used as the programming language of choice. This was supported by CLIPS (an expert system shell) that were used to trigger rules generated. The details of the databases as well as the various functions are presented in the following sub-sections.

6.1.2. Databases

Two databases, namely past design database and discernibility matrix database, are used to support the various functions of the prototype CDA. The databases were simulated using Microsoft Excel spreadsheets. The past design database stored all the past designs according to the representation scheme presented in Chapter 3. Past design database consisted of a set of columns (for design attributes) and rows (for design objects). The discernibility matrix database embodied the two global discernibility matrices described in Section 5.4.1.3 for design analysis and design synthesis respectively. It could be represented by a set of information tables. Each information table was meant for a decision attribute. On an information table, each of the cells represented the condition attributes that discern the pair of objects.

6.1.3. Updating of Past Design

The function, updating past design, is shown in Figure 6.1. It can be accessed via the past design update interface (Figure 6.2).

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

Figure 6.2. Past Design Update Interface

The past design update interface basically provides an avenue for the designers to modify a past design. New design attributes can be added and their values can also be changed. The past design update module, once invoked, enables the designers to retrieve and store past designs into the past design database.

6.1.4. Updating of Global Discernibility Matrices

The function, updating of global discernibility matrices, can be accessed via the global discernibility matrix update interface. Upon activation, the global discernibility matrix checking module examines the date of last updates for the past design database and the global discernibility matrix database so as to identify the cells in the global discernibility matrix database that need to be updated. The global discernibility matrix update module then modifies the entry of these cells based on the information retrieved from the past design database.

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

6.1.5. Design Analysis

The design analysis function can be accessed via the concept input interface (Figure 6.3). Here, the designers can specify the design concepts and the capabilities that need to be mapped. Using the method mentioned in Section 5.2.1, the temporary discernibility matrix retrieval module proceeds to identify those past design objects that are relevant. Subsequently, it retrieves the information stored in the global discernibility matrix database which corresponds to these past design objects. The information obtained is then saved in a temporary discernibility matrix which is used for rule generation by the rule construction module. Upon completion, the rule set generated is used by the rule trigger module for the estimation of design capabilities. The estimated design capabilities can be presented to the designers via the capability display interface (Figure 6.4).

Figure 6.3. Concept Input Interface

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

Figure 6.4. Capability Display Interface

6.1.6. Design Synthesis

The design synthesis function is fairly similar to that of design analysis. It can be accessed using the specification input interface (Figure 6.5), where the designers can specify design specifications and design concepts.

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

Figure 6.5. Specification Input Interface

The temporary discernibility matrix retrieval module is used to solicit past design objects that are relevant. The information stored in the global discernibility matrix database, which is associated with the past design objects, is first retrieved. The criterion for retrieval is based on the similarity measure mentioned in Section 5.2.1. The information is then stored as a temporary discernibility matrix, which is accessed by the rule construction module for rule generation. The rules obtained are used by the rule trigger module to derive design concepts, which are presented to the designers via the concept display interface (Figure 6.6).

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

Figure 6.6. Concept Display Interface

6.2. Case Study 1: Elastomer Mold Design

6.2.1. Descriptions

Case study 1 concerned the design of elastomer molds by a mold maker. Based on the discussions with the designers of elastomer molds and the sample mold designs provided by the mold maker, a set of design information comprising product variables (such as product price, general size and aspect ration) and mold variables (such as mold type, finishing and orientation) and design rules were obtained. The design information solicited can be represented using PBS representation as shown in Figure 6.7.

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

Figure 6.7. PBS Representation of Elastomer Mold Design

The PBS representation here did not contain any behaviour elements. Behaviour elements, were not probed during the interview as the case study aimed at establishing the relationship between the structure domain (mold variables) and the purpose domain (product variables). This information, i.e. the design and its constraints, can be described by a set of customer or product variables shown in Table 6.2.

In the case study, product variables and mold variables were treated as design concepts and design capabilities respectively. An elastomer mold design process typically begins with a customer presenting the design of an elastomer product accompanied by a list of design constraints. More specifically, the design can be described by its price range, the general product size, its aspect ratio, whether or not undercut/core and overmold are required, its material hardness, part tolerance, surface finish, and whether or not gating mark is allowed. Subsequently, the designers have to synthesise a mold design that can be used to manufacture the elastomer products. The most important decision to be made is to determine the mold type to be used. Three mold types, namely transfer mold,

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

compression mold and injection mold, are supported by the manufacturer. The selection of mold type has a large impact on the finished elastomer products. Other important decisions to be made are the finishing process required to be carried out on the mold, the usage of inserts, mold orientation, mold tolerance and also the number of cavities.

Table 6.2. A Summary of Elastomer Mold Design Variables

Customer/Product Variables				
Product Price	Low	Medium	High	
Product General size	Small (<10mm)	Medium (10-100mm)	Large (>100mm)	
Product Aspect ratio	Normal/squat	Flat/thin wall		
Product Undercut/core	Yes	No		
Product Overmold	Yes	No		
Product Material Hardness	Low (40A-50A)	High (60A-70A)		
Product Tolerance	Tight (<0.1mm)	Medium (0.1-0.25mm)	Loose (>0.25mm)	
Product Surface Finish	Mirror	Normal	Textured	
Gating Mark Allowed	Yes	No		
Designer/Mold Variables				
Mold Type	Transfer	Compression	Injection	
Mold Finishing	Normal	EDM	Chrome	Manual Polish
Mold Insert	Yes	No		
Mold Orientation	Any	Horizontal		
Mold Tolerance	Tight (<20um)	Medium (20-100um)	Loose (>100um)	
Mold No. of Cavities	Low	High		

As already mentioned, based on the interview with the mold designers, a set of design rules (Appendix 1) and 33 sets of mold design data (Appendix 2) were obtained. The mold design data were entered into the prototype CDA for rule generation. As for the design rules gleaned from interviewing the designers, they were used for comparison. Subsequently, the prototype CDA was used to perform design synthesis. Five new molds, that satisfied a set of product specifications were to be conceptualised. Using the

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

approaches described in Chapters 4 and 5, the design synthesis function was invoked to generate a set of design rules. These design rules were then used to generate design concepts to meet the product specifications.

6.2.2. Results and Discussions

A complex rule set was generated using the 33 mold design data. The sample rules after simplification are listed as follows.

- *If Product Size = Large then Mold No of Cavities = Low*
- *If Product Size = Small then Mold No of Cavities = High*
- *If Product Surface Finish = Normal then Mold Finishing = Normal*
- *If Product Surface Finish = Textured then Mold Finishing = EDM*
- *If Product Surface Finish = Mirror and Product Size = Small then Mold Finishing = Chrome*
- *If Product Undercut/Core = No then Mold Insert = No*
- *If Product Undercut/Core = Yes then Mold Insert = Yes*
- *If Product Aspect = Flat/Thin Wall then Mold Orientation = Horizontal*
- *If Product Aspect = Normal/Squat then Mold Orientation = Any*
- *If Product Tolerance = Tight then Mold Tolerance = Tight*
- *If Product Tolerance = Medium then Mold Tolerance = Medium*
- *If Product Price = Low then Mold Type = Compression*
- *If Product Tolerance = Loose then Mold Type = Compression*
- *If Product Price = Medium then Mold Type = Transfer*
- *If Product Price = High AND Product Tolerance = Medium then Mold Type = Injection*
- *If Product Price = High AND Product Tolerance = High then Mold Type = Injection*

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

- *If Product Hardness = High then Mold Type = Injection*
- *If Product Finishing = Mirror AND Product Size = Medium then Mold Finish = Polish*
- *If Product Finishing = Mirror AND Product Size = Large then Mold Finish = Polish*
- *If Product Tolerance = Medium then Mold Type = Transfer*
- *If Product Size = Medium then Mold No. of Cavities = Low*

The above-mentioned design rules agreed favourably with those gleaned from the mold designers. For example, the rules '*If Product Undercut/Core = No then Mold Insert = No*' and '*If Product Price = Low then Mold Type = Compression*' can be found in the set of design rules provided by the designers. This shows that the prototype CDA is able to discover reasonable design rules from a set of data. It is important to note that it is not possible or very difficult to derive these design rules by examining the data manually.

Subsequently, given a set of design capabilities, the rule set generated was used to synthesise the design concepts. The results of the synthesis are shown in Table 6.3. The case study shows that the design concept-capability mapping is applicable to design synthesis. From Table 6.3, about 80% of the design concepts generated were found to be valid after consulting the designers. For example, all the design concepts of New Molds 1 and 5 were found to be valid. As for New Molds 2, 3 and 5, only part of the design concepts were in order. More specifically, Mold Type and Number of Cavities for New Mold 2 and Mold Orientation and Number of Cavities for New Molds 3 and 4 were found to be incorrectly inferred. However, it was noted that most of the design concepts generated for the design of New Molds 2, 3 and 5 met the requirements. For example, for New Mold 3, out of six design concepts, five namely Mold Type, Finishing, Insert and Mold Tolerance were estimated correctly.

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

Table 6.3. Summary of Molding Design Synthesis

Customer/Product Variables	New Mold 1	New Mold 2	New Mold 3	New Mold 4	New Mold 5
Price	Low	Medium	High	Low	Medium
General size	Small	Medium	Large	Medium	Small
Aspect ratio	Normal/squat	Flat/thin wall	Normal/squat	Normal/squat	Flat/thin wall
Undercut/core	Yes	No	No	No	Yes
Overmold	Yes	No	Yes	Yes	No
Part Tolerance	Tight	Medium	Tight	Medium	Loose
Material Hardness	Low	High	Low	Low	High
Surface finish	Mirror	Normal	Normal	Normal	Textured
Gating marks allowed	Yes	No	No	Yes	No
Designer/Mold Variables	New Mold 1	New Mold 2	New Mold 3	New Mold 4	New Mold 5
Mold Type	Transfer	Compression	Injection	Compression	Transfer
Finishing	Manual Polish	Normal	Normal	Normal	EDM
Insert	Yes	No	No	No	Yes
Mold Orientation	Any	Horizontal	Horizontal	Horizontal	Horizontal
Mold Tolerance	Tight	Medium	Tight	Medium	Medium
No. of Cavities	High	High	High	High	High

Note: The designer/mold variables written in **bold** do not satisfy the design rules from designers.

Further analysis revealed that the design concepts of all New Molds, which were derived based on Attributes, Finishing, Inserts and Mold Tolerance, were estimated correctly using the rule set obtained. This could be attributed to useful design rules such as ‘If Product Undercut/Core = No then Mold Insert = No’ and ‘If Product Undercut/Core = Yes then Mold Insert = Yes’ had been derived. This, however, did not happen to all of the attributes such as Mold Type, Mold Orientation and Number of Cavities. Although only 33 objects were used in this case study, design rules with reasonable accuracy were obtained. The accuracy of the design rules in generating suitable design concepts can be improved if more past design data are made available for rule extraction. Once more past design data are made available, the global

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

discernibility matrix database as well as design rules can be updated using the approach described in Section 5.4.1.3.

6.3. Case Study 2: Cellular Phones Design

6.3.1. Descriptions

Case study 2 concerned the design of cellular phones by a multi-national manufacturer. Based on the historical data for the benchmarking of different types of cellular phones, a few salient design attributes were identified. These design attributes are listed as follows.

- Dimensions: The overall dimensions of the phone,
- Weight: The overall weight of the phone,
- Shape: The general shape of the phone (block, clam, PDA, etc),
- Antenna: Type of the antenna (internal, external or extensible),
- Screen size: Area of the screen,
- Talk time: Duration the battery can last when the phone is used for talking,
- Standby time: Duration the battery can last when the phone is on standby,
- Camera resolution: Resolution of the camera (if any),
- FM radio enable: Availability of FM radio,
- Loudspeaker enable: Availability of loudspeaker,
- MP3 player enable: Availability of MP3 player,
- Bluetooth connectivity: Ability to use bluetooth network,
- GPRS connectivity: Ability to use GPRS network, and
- Infrared connectivity: Ability to use infra red data transfer.

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

These attributes were chosen, as they have an impact on the acceptability of a cellular phone by customers. Accordingly, 4 customer groups or types were identified. They are listed as follows.

- No frill: Characterised by simple working features,
- Fancy: Characterised by devices that have some nice and novel features,
- Pro: Characterised by sophisticated and useful features, and
- Tech: Characterised by state-of-the-art features.

Basically, design attributes and the acceptability of a cellular phone by customers could be used to denote design concepts and design capabilities of the product respectively. For each customer group, three levels of acceptability, namely 'Unacceptable', 'Acceptable' and 'Excessive', were adopted. 'Unacceptable' indicated that the design of the cellular phone was less than desirable, whereas 'Acceptable' implied that the cellular phone was acceptable. 'Excessive' suggested that the cellular phone was too sophisticated for a particular group of customers. For 'No Frill' customers, the acceptability, 'Unacceptable', was not applicable, as this group of customers liked cellular phones without sophisticated features. As for 'Tech' customers, the acceptability, 'Excessive', was not needed, as they preferred the highest level of sophistication. Forty-four (44) different types of cellular phone data (Appendix 3) were gathered from the manufacturer's past design database. Subsequently, based on the data collected and through interview, a PBS model for the cellular phones was constructed (Figure 6.8).

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

Figure 6.8. PBS Representation for the Cellular Phones

Figure 6.8 shows a simplified conceptual representation for the design of a cellular phone. The representation is at the abstraction level, which is suitable for this case study. For example, the main function of the cellular phone, *communicate with others*, was associated with the behaviour element, *voice and SMS sending mechanism*, which was not elaborated further. In the structure domain, the main components of the cellular phone were *Main circuit*, *Antenna*, *Battery*, *Screen*, *MP3 player*, *Camera*, *Loudspeaker*, *Radio* and *Data transfer tools*. Many of these main components, in reality, are physically embedded in the main circuitry itself. However, functionally, they serve to accomplish particular functions.

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

These 44 sets of cellular phone design data (Appendix 3) were entered into the past design database. They were then used to assist the designers in analysing a new cellular phone design that was targeted at a particular customer group. The results of the analysis could then be used by the designers for subsequent designs.

To ascertain the performance of the prototype CDA, a series of 'leave-one-out' tests (Reich and Barai, 1999) was carried out. Using the 'leave-one-out' test, the design data of one of the cellular phones were removed. The remaining 43 sets of cellular phone design data were retained and used by the prototype CDA for the generation of design rules. The set of cellular phone design data that was removed from the test was then used to ascertain the validity of the design rules derived. More specifically, during the 'leave-one-out' test, the value attained by the condition attributes (design concepts) of the test data was used as inputs to prototype CDA for the estimation of the value of the decision attribute (design capabilities). The estimation obtained was then benchmarked against the actual values of the decision attributes of the test data. Subsequently, the 'leave-one-out' test was repeated for other design data.

6.3.2. Results and Discussions

The results of the complete 'leave-one-out' tests are summarised in Table 6.4a to 6.4d. In general, correlating 14 design attributes with the perception of customer groups is a daunting task, as no clear relationship can be derived by manually examining 44 sets of past design data. Using the prototype CDA, the aforementioned relationship can be easily extracted.

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

From Tables 6.4a to 6.4d, it is apparent that about 75% of the capabilities were estimated correctly. It is a well-known fact that product design is a creative process. Hence, varied perceptions about a design by different designers or design project teams are expected. This may have an impact on the estimation of capabilities.

Table 6.4a. Case Study 2 Design Analysis Results for No Frill Customer

Object Number	No Frill			Object Number	No Frill		
	Actual Capability	Estimated Capability	Reliability		Actual Capability	Estimated Capability	Reliability
1	Acceptable	Acceptable	53%	23	Excessive	Excessive	72%
2	Acceptable	Acceptable	75%	24	Excessive	Excessive	88%
3	Excessive	Excessive	75%	25	Excessive	Excessive	73%
4	Excessive	Excessive	89%	26	Excessive	Excessive	100%
5	Acceptable	Excessive	69%	27	Excessive	Excessive	100%
6	Excessive	Excessive	74%	28	Excessive	Excessive	100%
7	Acceptable	Acceptable	100%	29	Excessive	Excessive	75%
8	Excessive	Excessive	90%	30	Acceptable	Excessive	72%
9	Excessive	Excessive	86%	31	Acceptable	Acceptable	100%
10	Acceptable	Excessive	53%	32	Excessive	Excessive	81%
11	Acceptable	Excessive	68%	33	Acceptable	Excessive	71%
12	Excessive	Excessive	72%	34	Excessive	Excessive	77%
13	Excessive	Excessive	83%	35	Excessive	Excessive	90%
14	Excessive	Excessive	84%	36	Excessive	Excessive	100%
15	Acceptable	Excessive	78%	37	Excessive	Excessive	75%
16	Excessive	Excessive	81%	38	Excessive	Excessive	62%
17	Excessive	Acceptable	59%	39	Excessive	Excessive	96%
18	Excessive	Excessive	59%	40	Excessive	Excessive	96%
19	Excessive	Excessive	79%	41	Excessive	Excessive	100%
20	Excessive	Excessive	92%	42	Excessive	Excessive	96%
21	Excessive	Excessive	100%	43	Excessive	Excessive	90%
22	Excessive	Excessive	83%	44	Excessive	Excessive	96%

Note:
The **bold** entries under Estimated Capability indicate that the estimation was not the same as the Actual Capability

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

Table 6.4b. Case Study 2 Design Analysis Results for Fancy Customer

Object Number	Fancy			Object Number	Fancy		
	Actual Capability	Estimated Capability	Reliability		Actual Capability	Estimated Capability	Reliability
1	Unacceptable	Unacceptable	59%	23	Unacceptable	Unacceptable	67%
2	Unacceptable	Unacceptable	79%	24	Acceptable	Unacceptable	52%
3	Acceptable	Acceptable	52%	25	Acceptable	Unacceptable or Acceptable	48%
4	Acceptable	Acceptable	71%	26	Acceptable	Acceptable	100%
5	Unacceptable	Unacceptable	62%	27	Acceptable	Acceptable	100%
6	Acceptable	Acceptable	53%	28	Acceptable	Acceptable	82%
7	Acceptable	Acceptable	100%	29	Excessive	Acceptable	50%
8	Acceptable	Acceptable	47%	30	Unacceptable	Acceptable	53%
9	Acceptable	Acceptable	100%	31	Acceptable	Acceptable	100%
10	Unacceptable	Unacceptable	57%	32	Acceptable	Acceptable	65%
11	Unacceptable	Unacceptable	58%	33	Unacceptable	Unacceptable	50%
12	Excessive	Acceptable	56%	34	Unacceptable	Unacceptable	67%
13	Acceptable	Acceptable	54%	35	Unacceptable	Unacceptable	64%
14	Unacceptable	Unacceptable	61%	36	Acceptable	Unacceptable	94%
15	Acceptable	Unacceptable	70%	37	Unacceptable	Unacceptable	58%
16	Acceptable	Acceptable	64%	38	Acceptable	Acceptable	73%
17	Unacceptable	Unacceptable	50%	39	Acceptable	Acceptable	55%
18	Unacceptable	Unacceptable	78%	40	Acceptable	Acceptable	81%
19	Acceptable	Unacceptable	45%	41	Excessive	Acceptable	81%
20	Acceptable	Acceptable	60%	42	Unacceptable	Acceptable	48%
21	Unacceptable	Unacceptable	59%	43	Acceptable	Unacceptable	62%
22	Unacceptable	Acceptable	54%	44	Excessive	Acceptable	50%

Note:
The **bold** entries under Estimated Capability indicate that the estimation was not the same as the Actual Capability

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

Table 6.4c. Case Study 2 Design Analysis Results for Pro Customer

Object Number	Pro			Object Number	Pro		
	Actual Capability	Estimated Capability	Reliability		Actual Capability	Estimated Capability	Reliability
1	Unacceptable	Acceptable	58%	23	Excessive	Acceptable	58%
2	Unacceptable	Unacceptable	70%	24	Acceptable	Acceptable	55%
3	Excessive	Acceptable	44%	25	Excessive	Excessive	50%
4	Excessive	Excessive	58%	26	Acceptable	Acceptable	100%
5	Unacceptable	Acceptable	59%	27	Acceptable	Acceptable	100%
6	Excessive	Unacceptable or Acceptable	37%	28	Acceptable	Acceptable	90%
7	Acceptable	Acceptable	100%	29	Acceptable	Acceptable	70%
8	Acceptable	Acceptable	79%	30	Acceptable	Acceptable	47%
9	Acceptable	Acceptable	72%	31	Acceptable	Acceptable	100%
10	Unacceptable	Unacceptable	47%	32	Excessive	Excessive	50%
11	Unacceptable	Acceptable	54%	33	Acceptable	Acceptable	63%
12	Excessive	Acceptable	50%	34	Acceptable	Acceptable	53%
13	Excessive	Excessive	42%	35	Acceptable	Acceptable	68%
14	Unacceptable	Acceptable	57%	36	Acceptable	Acceptable	93%
15	Excessive	Unacceptable or Acceptable	35%	37	Unacceptable	Acceptable	43%
16	Acceptable	Acceptable	65%	38	Unacceptable	Excessive	80%
17	Acceptable	Acceptable	72%	39	Excessive	Excessive	45%
18	Unacceptable	Acceptable	67%	40	Unacceptable	Excessive	56%
19	Acceptable	Acceptable	63%	41	Excessive	Excessive	80%
20	Excessive	Acceptable	68%	42	Acceptable	Acceptable	68%
21	Unacceptable	Acceptable	77%	43	Acceptable	Acceptable	71%
22	Unacceptable	Acceptable	48%	44	Acceptable	Acceptable	73%

Note:
The **bold** entries under Estimated Capability indicate that the estimation was not the same as the Actual Capability

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

Table 6.4d. Case Study 2 Design Analysis Results for Tech Customer

Object Number	Tech			Object Number	Tech		
	Actual Capability	Estimated Capability	Reliability		Actual Capability	Estimated Capability	Reliability
1	Unacceptable	Unacceptable	91%	23	Unacceptable	Unacceptable	78%
2	Unacceptable	Unacceptable	100%	24	Acceptable	Unacceptable	68%
3	Unacceptable	Unacceptable	83%	25	Acceptable	Unacceptable	69%
4	Unacceptable	Unacceptable	97%	26	Unacceptable	Unacceptable	100%
5	Unacceptable	Unacceptable	87%	27	Unacceptable	Unacceptable	100%
6	Unacceptable	Unacceptable	90%	28	Acceptable	Unacceptable	76%
7	Unacceptable	Unacceptable	100%	29	Unacceptable	Unacceptable	88%
8	Unacceptable	Unacceptable	97%	30	Unacceptable	Unacceptable	86%
9	Unacceptable	Unacceptable	88%	31	Unacceptable	Unacceptable	100%
10	Unacceptable	Unacceptable	94%	32	Unacceptable	Unacceptable	88%
11	Unacceptable	Unacceptable	83%	33	Unacceptable	Unacceptable	88%
12	Acceptable	Acceptable	71%	34	Unacceptable	Unacceptable	90%
13	Unacceptable	Unacceptable	55%	35	Unacceptable	Unacceptable	93%
14	Unacceptable	Unacceptable	83%	36	Unacceptable	Unacceptable	97%
15	Acceptable	Unacceptable	83%	37	Unacceptable	Unacceptable	87%
16	Acceptable	Unacceptable	65%	38	Unacceptable	Unacceptable	88%
17	Unacceptable	Unacceptable	90%	39	Acceptable	Unacceptable	58%
18	Unacceptable	Unacceptable	93%	40	Unacceptable	Unacceptable	56%
19	Unacceptable	Unacceptable	86%	41	Acceptable	Acceptable	71%
20	Unacceptable	Unacceptable	87%	42	Unacceptable	Unacceptable	83%
21	Unacceptable	Unacceptable	87%	43	Acceptable	Unacceptable	75%
22	Acceptable	Unacceptable	72%	44	Acceptable	Unacceptable	56%

Note:
The **bold** entries under Estimated Capability indicate that the estimation was not the same as the Actual Capability

The accuracy of estimation can be further improved if more past design data are made available. Nevertheless, the rules extracted can help designers in ascertaining whether a cellular phone, which is still being conceptualised, is able to satisfy the targeted customer groups. The case study has also demonstrated that the prototype CDA is able to tackle all the issues discussed in Chapter 4, namely multiple decision attributes, incomplete information and non-discrete valued attributes.

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

The application of the PBS representation showed that it was able to capture both functional design purposes such as *transfer data*, *capture picture* and *play MP3* and non-functional design purpose such as *user satisfaction*. For example, in this case study, the ultimate purpose was to achieve user satisfaction, which was represented as a non-functional design purpose. Such a non-functional design purpose was decomposed into several functional design purposes. In other words, the achievement of these functional design purposes would, in turn, contributed to the realisation of a higher level design purpose, *user satisfaction*. The functional design purposes were related to the working mechanisms in the behaviour domain. These mechanisms were not elaborated further in this case study, as it was not necessary to examine the internal working mechanisms to achieve the functions for the purpose of this case study. Subsequently, these working mechanisms were related to their respective design concepts or components in the structure domain. Besides, several design concepts in the structure domain were also correlated with the design purpose, *user satisfaction*. For instance, the attributes, *overall weight*, *overall size* and *shape*, were linked directly with the design purpose, *user satisfaction*. This implied that these variables also had a direct impact on the realisation of the design purpose, *user satisfaction*. It should be noted that these attributes did not have any tangible functions but they were important considerations for the realisation of the design purpose. In other words, the PBS representation scheme provides a means for designers to visualise the contribution of the various design attributes in relation to the realisation of a design purpose.

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

6.4. Summary

The proposed approaches described in Chapters 3 and 4 and the framework presented in Chapter 5 have been implemented as a prototype Conceptual Design Assistant (CDA) to assist designers in performing two main functions, namely design synthesis and design analysis. The prototype CDA comprises 2 databases, 4 user interfaces and 9 modules. It is able to support 4 functions, namely updating of past design (subsidiary function 1), updating of global discernibility matrices (subsidiary function 2), design analysis (main function 1) and design synthesis (main function 2). The two databases are namely past design database and global discernibility matrix database respectively. The function, updating of past design, is realised via the past design update interface and the past design update module. As for the function, updating of global discernibility matrices, it is achieved through the global discernibility matrix update interface, the global discernibility matrix checking module and the global discernibility matrix update module. The main function, design synthesis, is realised via the concept input interface, the temporary discernibility matrix retrieval module, the rule construction module, the rule trigger module and the capability display interface. The other main function, design analysis, is achieved via the specification input interface, the temporary discernibility matrix retrieval module, the rule construction module, the rule trigger module and the concept display interface.

The capability of the prototype CDA had been illustrated using two case studies. In both case studies, the PBS representation scheme was used to design of the products. The first case study concerned design synthesis of elastomer molds. From the case study, it had been established that the prototype system was able to generate reasonable design

CHAPTER 6 SYSTEM IMPLEMENTATION AND CASE STUDIES

rules using only 33 sets of past design data. Indeed, the design rules extracted agreed favourably with those gleaned from the expert designers. As for the design synthesis function, the prototype CDA was used to generate possible design concepts based on the inputs of the designers. The case study revealed that based on the design rules extracted, the prototype CDA was able to generate sensible design concepts with 80% agreement with the actual design concepts perceived by the designers. The second case study concerned the design of cellular phone. It had been shown that the prototype CDA was able to assist designers in carrying out the function of design analysis. Based on the inputs of the designers, the prototype CDA was able to suggest the acceptability of a new cellular phone design in different customer groups with reasonable accuracy. To further ascertain the reliability of the prototype CDA, a complete series of 'leave-one-out' tests was carried out. The test results showed that the prototype CDA was able to generate reasonable estimation with 75% accuracy.

CHAPTER 7 CONCLUSIONS AND FUTURE WORK

Chapter 7 Conclusions and Future Work

7.1. Conclusions

Product design is an important part of product development, as it commits about 80% of the total product costs. As products become more sophisticated due to the availability of newer and more advanced technologies, multi-disciplinary knowledge is needed for the product development. Furthermore, increasing product complexity, shorter product development life cycle and globalisation have resulted in a paradigm shift in product development, from single-organisation effort to multi-organisation collaborative effort, viz. collaborative design. This work aims at developing a prototype system to assist designers in conceiving (generation of potential design concepts) and selecting potential design concepts (estimation of design capabilities) during the conceptual design stage.

From the literature review, it is apparent that companies begin to form design chains in order to remain competitive. A more comprehensive design representation scheme is therefore needed to facilitate the communication among the stakeholders. It has been established that existing functional representation schemes, although promising, are not adequate to deal with this changing trend. For this purpose, the Purpose-Behaviour-Structure (PBS) representation scheme for conceptual design is proposed and implemented in this work. To facilitate the retrieval of relevant past design data represented in PBS, a similarity measure is proposed and implemented.

The formation of design chain also causes a shift in the way in which design is carried out. Design specifications are dynamically revised as the design process progresses. As a result, early detection of inadequacy in design due to changing design specifications

CHAPTER 7 CONCLUSIONS AND FUTURE WORK

becomes inevitable. For this purpose, the notion of design concept-capability mapping is proposed and implemented in this work to assist designers in assessing the impact of design changes and suggesting alternative design concepts. The various artificial intelligence techniques, such as knowledge based systems, artificial neural network and rough set theory, have been reviewed. It has been established that the rough set theory possesses excellent potential for the realisation of the proposed design concept-capability mapping. However, direct application of rough set to design concept-capability mapping is hindered by four issues, namely handling incomplete information, treating of non-discrete valued attributes, managing multiple decision attributes and incremental updating of rules. These issues provide the motivation for the author to develop a hybrid approach to tackle them.

More specifically, the proposed PBS representation scheme enables both functional and non-functional design purposes of design concepts to be communicated among the stakeholders in a simple, flexible and yet effective way. Based on the representation scheme, a framework for collaborative product development has also been proposed. From the case study on vacuum cleaner design, it has been ascertained that the proposed PBS representation scheme is able to capture both functional and non-functional design purposes effectively. Furthermore, such a representation scheme enables functional and non-functional purposes to be traced to their corresponding structure elements or attributes.

As for the proposed design concept-capability mapping, it uses a set of design rules, which is extracted from past design data to estimate the expected performance. Given a set of design concepts, the mapping, which is represented by a set of design rules, can

CHAPTER 7 CONCLUSIONS AND FUTURE WORK

be used to estimate the expected performance. A detailed study has been carried out to examine the above-mentioned issues, i.e. handling incomplete information, treating of non-discrete valued attributes, managing multiple decision attributes and incremental updating of rules. Accordingly, four approaches have been established to handle these issues.

Specifically, a so-called attributes decomposition algorithm has been developed to tackle the issue concerning multiple decision attributes. The attributes decomposition algorithm splits an information system with multiple decision attributes into a number of information systems with single decision attribute. Rules are then derived from each of these information systems, and subsequently amalgamated using Boolean algebra. The validity of the attributes decomposition algorithm has been proven mathematically. The attributes decomposition algorithm offers a practical solution to simplify the rule derivation process.

As for the issue concerning incomplete information, it can be tackled using a so-called dissimilar objects algorithm. Basically, incomplete information concerns the usage of additional past design data with partial similarity with existing ones. In so doing, incomplete information is introduced into the information system and results in a larger search space for rule extraction. Based on the dissimilar objects algorithm, 'UA' values, i.e. unavailable, are used to substitute the missing values in an information system. Once the rules have been extracted, those rules with 'UA' values are eliminated. It has been shown mathematically that the rules extracted by the proposed dissimilar objects algorithm are more precise.

CHAPTER 7 CONCLUSIONS AND FUTURE WORK

The issue of non-discrete valued attributes emerges from the fact that in design, the values attained by design attributes can be continuous-valued or discrete-valued. While discrete valued attributes can naturally be handled by the rough set theory, continuous valued attributes cannot. In this work, the MD-Heuristic algorithm has been extended to deal with this issue. The extended MD-Heuristic algorithm adopts design specifications as the basis for the discretisation of the continuous valued decision attributes. As a result, an information system containing fully discrete valued attributes can be obtained.

It has been established that the evolution of the design of a product may bring about changes in the past design data extracted from the past design database, and hence the design rules. An approach to incremental updating of rules becomes necessary. A detailed analysis on the various ways in which incremental updating of rules has been carried out. Based on the study, a novel knowledge representation scheme that makes use of a so-called global discernibility matrix is proposed and implemented. Incremental updating of rules can be achieved by 1) updating of the global discernibility matrix, 2) extracting relevant information from the updated global discernibility matrix to form a so-called temporary discernibility matrix, and 3) extracting rules using the temporary discernibility matrix.

The above mentioned work has established the theoretical basis for the realisation of a prototype Conceptual Design Assistant (CDA). An overall framework of the CDA, which comprises 6 modules, namely design concept-capability mapping mechanism, past design retriever, GUI – inputs for specifications and feedback, design capability assessor, design concept extractor, and incremental updater, has been developed. Based on this framework and the aforementioned approaches, a prototype CDA has been

CHAPTER 7 CONCLUSIONS AND FUTURE WORK

realised to assist designers in performing two main functions, namely design synthesis and design analysis. The capability of the prototype CDA has been illustrated using two case studies. It has been shown that the aforementioned PBS representation scheme can be used to describe the design of a product effectively. From the first case study that concerned the design of elastomer molds, it had been established that the prototype CDA was able to generate reasonable design rules using a small amount of past design data. The design rules extracted agreed favourably with those gleaned from the expert designers. As the case study concerned design synthesis, the results showed that based on the design rules extracted, the prototype CDA was able to generate sensible design concepts with good (80%) agreement with the actual design concepts perceived by the designers. The second case study that concerned the design of cellular phones was used to illustrate the ability of the prototype CDA in carrying out design analysis. Similar to the first case study, the prototype CDA was able to extract sensible rules from past design data. Based on the design rules extracted and the inputs of the designers, the prototype CDA had demonstrated its ability in predicting the acceptability of a new cellular phone design by different customer groups with reasonable accuracy. From the results of a complete series of the so-called 'leave-one-out' tests, it was apparent that the prototype CDA was able to generate reasonable estimation with good (75%) accuracy.

The contribution of the work (Alisantoso et al., 2004a; Alisantoso et al. 2004b; Alisantoso et al., 2005a; Alisantoso et al., 2005b) undertaken by the author can be summarised as follows.

CHAPTER 7 CONCLUSIONS AND FUTURE WORK

- A novel design knowledge representation scheme, i.e. the Purpose-Behaviour-Structure representation scheme, for conceptual design,
- A reliability measure for past design retrieval,
- A hybrid approach, which is driven by the design rules gleaned from past design data, to design analysis and design synthesis,
- An algorithm to handle incomplete information (dissimilar objects algorithm),
- An algorithm to handle multiple decision attributes (attributes decomposition algorithm),
- An algorithm to handle non-discrete valued attributes (extended MD-heuristic algorithm),
- A novel rough set based approach to incremental updating of rules through a so-called global discernibility matrices,
- A framework and a prototype system known as Conceptual Design Assistant incorporating the aforementioned approaches.

7.2. Future Work

The work presented in this thesis has established the basic building blocks for the realisation of a comprehensive environment for collaborative design. Future work includes:

- i. *Establishment of a comprehensive scheme for product classification so as to facilitate past design retrieval*

In this work, the retrieval of relevant past design is based on similarity measure, which evaluates the degree of similarity between the current design task and past designs.

CHAPTER 7 CONCLUSIONS AND FUTURE WORK

Such an evaluation does not take into account the relative importance of the design attributes of past designs. As already discussed, the relevance of past design data that are used to form the global discernibility matrix or information system, has an impact on the quality of rules extracted. A comprehensive product classification scheme can be developed to facilitate arrangement of past designs into similar groups. In so doing, more and better past design data can possibly be retrieved. As a result, better and more reliable rules can be extracted. However, the scheme must not be too constricting, as it will limit the search space of possible solutions.

ii. Implementation of the CDA in a comprehensive collaborative environment

Currently, the CDA is implemented as a stand alone desktop system. The full potential and benefits of the CDA can be realised when it is implemented in a collaborative design environment. It is envisaged that in a collaborative design environment, past design data will be made available via a distributed database. More past design data will be available for rule extraction. As a result, the reliability of the design rules extracted would be improved. A uniform or compatible design knowledge representation scheme that can be accepted by all stakeholders is necessary. The PBS representation scheme proposed in this work is well poised to be one possible solution for such a purpose. However, issues such as intellectual property rights, ethics and security among the participating companies need to be resolved before any real collaboration can take place.

References

Alisantoso, D., Khoo, L.P., Lee, I.B.H., Fok, S.C. (2004a) A rough set approach to design concept analysis in design chain, *International Journal of Advanced Manufacturing Technology*, (published online).

Alisantoso, D., Khoo, L.P., Lee, I.B.H. (2004b) An approach to the analysis of design concepts using rough set theory, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 18, 343-355.

Alisantoso, D., Khoo, L.P., Lee, I.B.H., Lu, W.F. (2005a) A design representation scheme for collaborative product development, *International Journal of Advanced Manufacturing Technology*, (in print).

Alisantoso, D., Khoo, L.P., Lu, W.F. (2005b) Early design analysis using rough set theory, *Intelligent Production Machines and Systems (IPROMS)*, (to be held in July 2005).

Alsabti, K., Ranka, S. and Singh, V. (1998) CLOUDS: A decision tree classifier for large datasets, *Proceedings of KDD-98*, Edited by Agrawal, R., Storlorz, P. and Piatetsky-Shapiro G., 2-8.

Arbab, F. (1987) A paradigm for intelligent CAD, *Intelligent CAD Systems 1: Theoretical and Methodological Aspects*, Springer-Verlag.

Arbab, F. and Wing, J. (1985) Geometric reasoning: A new paradigm for processing geometric information, *IEEE International Symposium on New Directions in Computing*, August 1985.

Arnold, F. (2000) MACAO – A journey into CAX interoperability and collaborative design, *Proceedings of IEEE International Conference on Information Visualization*, 557-562, London, UK.

Arslan, A. and Ince, R. (1996) Neural network-based design of edge-supported reinforced concrete slabs, *Structural Engineering Review*, 8(4), 329-335.

Bain, W.M. (1989) Judge, *Inside Case-Based Reasoning*, Edited by Riesback, C.K. and Schank, R.C., 93-139, Lawrence Erlbaum Associates, New Jersey.

Baniassad, E.L.A., Murphy, G.C. and Schwanninger, C. (2003) Design pattern rationale graphs: linking design to source, *Software Engineering, Proceedings 25th International Conference on*, 352-362, 3-10 May 2003.

Barnes, C.J., Dalglish, G.F., Jared, G.E.M., Mei, H. and Swift, K.G (1999) Assembly oriented design, *Proceedings of IEEE International Symposium on Assembly and Task Planning*, 45 –50, Porto, Portugal.

Beale, R. and Jackson, T. (1990) *Neural Computing: An Introduction*, Institute of Physics Publishing, Bristol, UK.

Bellotti V. (1993) Integrating theoreticians' and practitioners' perspectives with design rationale, *Proceedings of INTERCHI'93*, 104-114. Amsterdam, Netherlands, April 24-25.

Bochenek, G.M. and Ragusa, J.M. (2001) Virtual collaborative design environments: A review, issues, some research and the future, *Portland International Conference on Management of Engineering and Technology*, Supplement, 726-735, Portland, USA.

Borner, K. (1998) CBR for design, *Case-Based Reasoning Technology: From Foundations to Applications*, Edited by Lenz, M., Burkhard, H.D., Bartsch-Sporl, B. and Wess, S., 201-231, Springer-Verlag, Berlin Heidelberg.

Calvanese, D., De Giacomo, G., Lenzerini, M., Nardi, D. and Rosati, R. (1998) Knowledge representation approach to information integration, *Proceedings of AAAI Workshop on AI and Information Integration*, 58-65, Madison, WI, July 26-27.

Chiu, M.L. (2003) Design moves in situated design with case-based reasoning, *Design Studies*, 24 (1), 1-25.

Chou, S., Chin, K.S., Yarlagadda, P.K.D.V. (2003) Internet-based intensive product design platform for product design, *Knowledge-based System*, 16, 7-15.

Collins, G.C. (1989) Plan creation. *Inside Case-Based Reasoning*, Edited by Riesback, C.K. and Schank, R.C., 249-289, Lawrence Erlbaum Associates, New Jersey.

Conklin, E. J. and Burgess-Yakemovic, K. C. (1996) A process oriented approach to design rationale. *Design Rationale: Concepts, Techniques and Use*, Edited by Moran, T.P. and Carroll, J. M., 393-427, Lawrence Erlbaum Associates, NJ.

Coyne, R.D., Rosenman, M.A., Radford, A.D. and Gero, J.S. (1987) Innovation and creativity in knowledge-based CAD, *Expert Systems in Computer Aided Design*, Edited by Gero, J.S., 435-465, North-Holland, Amsterdam.

Dai, W. and Drogemuller, R. (1999) Collaborative framework for building design, *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*, 4, 23-32, Tokyo, Japan.

Deng Y.M., Tor S.B. and Britton G.A (2000) Abstracting and exploring functional design information for conceptual mechanical product design, *Engineering with Computers*, 16, 36-52.

Edmonds, R.A. (1988) *The Prentice Hall Guide to Expert Systems*, Prentice-Hall, New Jersey

Felix, R. and Ushio, T. (1999) Rules induction from inconsistent and incomplete data using rough sets, *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*, 5, 154-158, Tokyo, Japan.

Fenves, S. J. and Baker, N. C. (1987) Spatial and functional representation language for structural design, *Expert Systems in Computer Aided Design*, Elsevier Science, North Holland.

Frost, P. and Warren, P. (2000) Virtual reality used in a collaborative design process, *Proceedings of IEEE International Conference on Information Visualization*, 568-573, London, England.

Fu, L.M. (1994) *Neural Networks in Computer Intelligence*, Mc-Graw Hill, Singapore.

Fung, A.H., Lai, Y.N. and Sheu, B.J. (1988) A knowledge-based approach to analog IC design, *IEEE Transactions on Circuits and Systems*, 35(2), 256 –258.

Gehrke, J., Ganti, V., Ramakrishnan, R. and Loh, W.Y. (1999) BOAT - Optimistic decision tree construction, *Proceedings of International Conference on Management of Data*, 169-180, Philadelphia, Pennsylvania, USA.

Gero, J. S. (1990) Design prototypes: a knowledge representation schema for design. *AI Magazine*, 11(4), 26-36.

Goel, A., Bhatta, S. & Stroulia, E. (1997) Kritik: An early case-based design system. *Issues and Applications of Case-Based Reasoning in Design*, Edited by Maher, M. and Pu, P., Mahwah, NJ: Erlbaum, 87-132.

Gomez de Silva Garza, A. and Maher, M.L. (1997) Case-based reasoning in design, *IEEE Expert*, 12(2), 34-41.

Grzymala-Busse, J.W. and Hu, M. (2000) A comparison of several approaches to missing attribute values in data mining, *Second International Conference on Rough Sets and Current Trends in Computing*, 340-347, Banff, Canada.

Grzymala-Busse, J.W. (1992) LERS: A system for learning from examples based on rough sets. *Intelligent Decision Support-Handbook of Applications and Advances of the Rough Sets Theory*, Edited by Slowinski, R., 3-18, Kluwer Academic Publishers, Dordrecht, Netherlands.

Hara, K. and Nakayama, K. (1996) Selection of minimum training data for generalization and on-line training by multilayer neural networks, *Proceedings of IEEE International Conference on Neural Networks*, 1, 436-441.

Helvacioğlu, S. and Insel, M. (2003) An expert system approach to container ship layout design, *International Shipbuilding Progress*, 50(1-2), 19-34.

Hicks, D.L. and Wiil, U.K. (2001) Towards collaborative design in an open hypermedia environment, *Sixth International Conference on Computer Supported Cooperative Work in Design*, 181-189, London, Ontario, Canada.

Hong, T.P. and Tseng, S.S. (1992) Comparison of ID3 and its generalized version, *Proceedings of Fourth International Conference on Computing and Information*, 241-244, Toronto, Canada.

Huang, H., Li, X., Cao, S. and Pan, W. (2001) The collaborative mechanical product design system, *Sixth International Conference on Computer Supported Cooperative Work in Design*, 163-168, London, Ontario, Canada.

Hundal, M.S. (1997), *Systematical Mechanical Designing: A Cost and Management Perspective*, The American Society of Mechanical Engineers.

Jackson, P.C. (1985) *Introduction to Artificial Intelligence*, Dover Publications, New York.

Jiang, B., Chen, C. and Bu, J. (2001) CoDesign - A collaborative pattern design system based on agent, *Sixth International Conference on Computer Supported Cooperative Work in Design*, 319–323, London, Ontario, Canada.

Jiye, L. and Zongben, X. (2000) Uncertainty measure of roughness of knowledge and rough sets in incomplete information systems, *Proceedings of the 3rd World Congress on Intelligent Control and Automation*, June 28-July 2, Hefei, P.R. China.

Joshi, M.V., Karypis, G. and Kumar, V. (1998) ScalParC: A new scalable and efficient parallel classification algorithm for mining large datasets, *12th International Parallel Processing Symposium*, 573-579, Orlando, Florida, USA.

Kang, H.T. and Yoon, C. J. (1994) Neural network approaches to aid simple truss design problems, *Microcomputers in Civil Engineering*, 9(3), 211-218.

Karsenty, L. (1996) Empirical evaluation of design rationale documents, *Proceedings of Conference on Human Factors in Computing Systems*, 150-156.

Khoo, L.P. and Zhai, L.Y. (2001) Multiconcept classification of diagnostic knowledge to manufacturing systems: Analysis of incomplete data with continuous-valued attributes, *International Journal of Production Research*, 39(17), 3941-3957.

Kolodner, J.L. (1993) *Case-Based Reasoning*, Morgan-Kaufmann Publishers, San Mateo, CA.

Komorowski, J., Pawlak, Z., Polkowski, L. and Skowron, A. (1999) Rough sets: A tutorial, *Rough Fuzzy Hybridization*, Edited by Skowron, A., 3-98, Springer, Singapore

Kryszkiewicz, M. (1998) Rough set approach to incomplete information systems, *Information Sciences*, 113, 39-49.

Kryszkiewicz, M. (1999) Rules in incomplete information systems, *Information Sciences*, 112, 271-292.

Kusiak, A., Kern, J.A., Kernstine, K.H. and Tseng, B.T.L. (2000) Autonomous decision-making: A data mining approach, *IEEE Transactions on Information Technology in Biomedicine*, 4(4), 274-284.

Lavana, H., Brglez, F., Reese, R., Konduri, G. and Chandrakasan, A. (2000) OpenDesign: An open user-configurable project environment for collaborative design and execution on the internet, *Proceedings of IEEE International Conference on Computer Design*, 567-570, Austin, Texas, USA.

Lee, J. and Lai, K.Y. (1991) What's in design rationale?, *Human-Computer Interaction*, 6(3-4), 251-280.

Lee, S. and Vachtsevanos, G (2002) An application of rough set theory to defect detection of automotive glass, *Mathematics and Computers in Simulation*, 60, 225-231.

Lee, Y.S., Liu, X.Z., Siu, K.W. and Wang, S.J. (1997) Expert system aided design, *23rd International Conference on Simulation and Optimization of Power Converters, Industrial Electronics, Control and Instrumentation*, 3, 1016 –1021, New Orleans, USA.

Lees, B. , Hamza, M. and Irgens, C. (2000) Case-based reasoning support for engineering design, *Proceedings of SPIE - The International Society for Optical Engineering*, 4192, 394-402.

Lenz, M., Auriol, E, and Manago, M. (1998) Diagnosis and decision support, *Case-based Reasoning Technology: From Foundations to Applications*, Edited by Lenz, M.,

Burkhard, H.D., Bartsch-Sporl, B. and Wess, S., 51-90, Springer-Verlag, Berlin Heidelberg.

Liang, J. and Xu, C. (2000) Uncertainty measures of roughness of knowledge and rough sets in incomplete information systems, *Proceedings of Third World Congress on Intelligent Control and Automation*, 2526-2529, Hefei, China.

Libardi, E.C., Dixon, J.R. and Simmons, M.K. (1988) Computer environments for the design of mechanical assemblies: A research review, *Engineering with Computers*, 121-136.

Liu, Y., Shi, Y and Xu, G. (2001) Supporting group awareness in collaborative design, *Sixth International Conference on Computer Supported Cooperative Work in Design*, 36-40.

MacGregor, S.P., Thomson, A.I and Juster, N.P. (2001) A case study on distributed, collaborative design: Investigating communication and information flow, *Sixth International Conference on Computer Supported Cooperative Work in Design*, 249-254, London, Ontario, Canada.

Martin, G. and Schirrmeister, F. (2002) A design chain for embedded systems, *IEEE Computer*, 35(3), 100-103.

Medsker, L. (1992) Neural computing and AI, *Expert System and Applied Artificial Intelligence*, Edited by Turban, E, 621-663, Macmillan Publishing Company, New York.

Miller, L.C.G. (1993) *Concurrent Engineering Design*, Society of Manufacturing Engineers.

Mollestad, T. and Komorowski, J. (1999) A rough set framework for mining propositional default rules, *Rough Fuzzy Hybridization*, Edited by Skowron, A., 233-262, Springer, Singapore.

Nakatani, Y., Tsukiyama, M. and Fukuda, T. (1991) Case Organization in a case-based engineering design support system, *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, 3, 1789-1794.

Nevins, J.L. and Whitley, D.E. (1989) *Concurrent Design of Products and Processes: A Strategy for the Next Generation in Manufacturing*, McGraw-Hills.

Newell, A. and Simon, H.A. (1972) *Human Problem Solving*, Prentice-Hall, New Jersey.

Nguyen, H.P., Le, L.P., Santiprabhob, P. and De Baets, B. (2001) Approach to generating rules for expert systems using rough set theory, *Joint 9th IFSA World Congress and 20th NAFIPS International Congress*, 2, 877-882, Vancouver, British Columbia, Canada.

Nguyen, L. and Swatman, P.A. (2000) Complementary use of ad hoc and post hoc design rationale for creating and organising process knowledge, *Proceedings of the Hawaii International Conference on System Sciences*, 50.

Nguyen, S.H. and Skowron, A. (1997) Quantization of real value attributes: Rough set and Boolean reasoning approach, *Bulletin of International Rough Set Society*, 1(1), 5-16.

Noelke, U. (1988) The essence of knowledge engineering, *Artificial Intelligence and Expert Systems*, Edited by Savory, S., 116-132, Ellis Horwood, England.

Øhrn, A. and Komorowski, J. (1997) ROSETTA: A rough set toolkit for analysis of data, *Proceedings of Third International Joint Conference on Information Sciences, Fifth International Workshop on Rough Sets and Soft Computing*, 3, 403-407, Durham, NC, USA.

Otto, K. and Wood, K. (2001) *Product Design*, Prentice-Hall, New Jersey.

Pahl G. and Beitz W. (1988) *Engineering Design: A Systematic Approach*, Springer-Verlag, Berlin.

Pawlak, Z. (1982) Rough sets, *International Journal of Computer and Information Sciences*, 11, 341-356.

Pawlak, Z. (1998) Granularity of knowledge, indiscernibility and rough sets, *Proceedings of IEEE World Congress on Computational Intelligence and IEEE International Conference on Fuzzy Systems*, 1, 106-110, Anchorage, Alaska, USA.

Pawlak, Z. (1996a) Rough sets and data analysis, *Proceedings of Asian Fuzzy Systems Symposium on Soft Computing in Intelligent Systems and Information Processing*, 1-6, Kenting, Taiwan.

Pawlak Z. (1996b) Why rough sets?, *Proceedings of Fifth IEEE International Conference on Fuzzy Systems*, 2, 738 –743, New Orleans, USA.

Qian L. and Gero J.S. (1996) Function-behaviour-structure paths and their role in analogy-based design, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 10, 289-312.

Quinlan, J.R. (1986) Induction of decision trees, *Machine Learning*, 1, 81-106.

Quinlan, J.R. (1993) *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo.

Quinlan, J.R. (2000) *Rulequest Website*, <http://www.rulequest.com/see5-comparison.html>

Shi, F., Lou, Z.L., Lu, J.G. and Zhang, Y.Q. (2003) An improved rough set approach to design gating scheme for injection moulding, *International Journal of Advanced Manufacturing Technology*, 21, 662-668.

Reich, Y. and Barai, S. V. (1999) Evaluating machine learning models for engineering problems, *Artificial Intelligence in Engineering*, 13, 257-272

Richter, H. A. and Abowd, G. D. (2000) Automating the capture of design knowledge, *22nd International Conference on Software Engineering*, Limerick, Ireland, June 7-9.

Ritcher, M.M. (1998) Introduction, *Case-Based Reasoning Technology: From Foundations to Applications*, Edited by Lenz, M., Burkhard, H.D., Bartsch-Sporl, B. and Wess, S., 1-15, Springer-Verlag, Berlin Heidelberg.

Riesback, C.K. and Schank, R.C. (1989) *Inside Case-Based Reasoning*, Lawrence Erlbaum Associates, New Jersey.

Rosenman M A and Gero J S. (1996) Modelling multiple views of design objects in a collaborative CAD environment, *Computer Aided Design*, 28(3), 193- 205, Elsevier Science Ltd.

Safoutin, M.J. and Smith, R.P. (1996) The iterative component of design, *IEEE International Conference on Engineering Management*, 564-569, Vancouver, British Columbia, Canada.

Savory, S. (1988) *Artificial Intelligence and Expert Systems*, Ellis Horwood, England.

Sembugamoorthy, V. and Chandrasekaran, B. (1986) Functional representation of devices and compilation of diagnostic problem solving system, *Experience, Memory and Reasoning*, Edited by Kolodner, J. and Riesback, C., 47-73, Lawrence Erlbaum, Hillsdale.

Sevy, J., Zaychik, V. Hewett, T.T. and Regli, W.C. (2000) Developing and evaluating collaborative engineering studios, *Proceedings of IEEE 9th International Workshops on Enabling Technologies: Infrastructure and Collaborative Enterprises*, 118-123, Gaithersburg, MD, USA.

Shan, S. and Wang, G.G. (2004) Space exploration and global optimization for computationally intensive design problems: A rough set based approach, *Struct Multidisc Optim*, 28, 427-441.

Shen, W. (2000) Web-based infrastructure for collaborative product design: An overview, *Proceedings of Fifth International Conference on Computer Supported Cooperative Work in Design*, 239-244, Hong Kong, China.

Shephard, M.S., Korngold, E.V. and Wentorf, R. (1999) *Idealized Models in Engineering Analysis*, Technical Report, Rensselaer Design Research Center, RPI.

Shiau, J.Y., Ratchev, S.M. and Valtchanov, G. (2000) Distributed collaborative design and manufacturability assessment for extended enterprise in XML-based agent system, *Proceedings of IEEE 9th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 260-265, Gaithersburg, MD, USA.

Song, P., Tang, M. and Dong, J. (2001) Collaborative model for concurrent product design, *Sixth International Conference on Computer Supported Cooperative Work in Design*, 212-217, London, Ontario, Canada.

Sturges, R.H. and Gero, J.S. (1996) Computational model for conceptual design based on function logic, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 10, 255-274.

Taludkar, S., Elfes, A., Oaoanikolopoulos, N., (1988) *Concurrent Design, Simultaneous Engineering and Distributed Problem Solving*, Technical Report, Engineering Design Center, Carnegie Mellon.

Tian, G.Y. and Taylor, D. (2001) Design and implementation of a web-based distributed collaborative design environment, *Proceedings of Fifth International Conference on Information Visualization*, 703-707, London, England.

Tong, L. and An, L. (2002) Incremental learning in decision rules based on rough set theory, *Proceedings of the 4th World Congress on Intelligent Control and Automation*, 420-425, Shanghai, P.R.China.

Turban, E. (1992) *Expert System and Applied Artificial Intelligence*, Macmillan Publishing Company, New York.

Tuthill, G.S. (1990) *Knowledge Engineering: Concepts and Practices for Knowledge-Based Systems*, Tab Books

Tversky, A. (1977) Features of similarity, *Psychological Review*, 84, 327-352.

Vargas-Hernandez, N., Shah, J.J and Lacroix, Z (2002) Knowledge representation for conceptual engineering design, *3rd International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, Madrid, Spain, June.

Ulrich, K.T. and Eppinger, S.D. (2000) *Product Design and Development*, Mc-Graw Hills, USA.

Umeda, Y., Ishii, M., Yoshioka, M., Shimomura, Y. and Tomiyama, T. (1996) Supporting conceptual design based on the functional-behaviour-state modeller, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 10, 275-288.

Umeda, Y. and Tomiyama, T. (1997) Functional reasoning in design, *IEEE Expert*, 12(2), 42-48.

Wang, G. (2002) Extension of rough set under incomplete information systems, *Proceedings of IEEE International Conference on Fuzzy Systems*, 2, 1098-1103, Honolulu, HI, USA.

Wang, Q. and Liao, W. (2004) Application and development of case-based reasoning in fixture design, *Journal of Southeast University (English Edition)*, 20(2), 170-175.

Winston, P.H. (1977) *Artificial Intelligence*, Addison-Wesley.

Wong, S.K.M. and Ziarko, W. (1986) INFER: An adaptive decision support system based on the probabilistic approximate classification, *Sixth International Workshop on Expert Systems and Their Applications*, 1, 713-726, Avignon, France.

Xie, H. (2001) Tracking of design changes for collaborative product development, *Sixth International Conference on Computer Supported Cooperative Work in Design*, 175 – 180, London, Ontario, Canada.

Yamakawa, T. (1987) Fuzzy hardware systems of tomorrow, *Approximate Reasoning in Intelligent Systems, Decisions and Control*, Edited by Sanchez, E. and Zadeh, L.A., 1-20, Pergamon Press, Oxford, England.

Zhai, L.Y., Khoo, L.P. and Fok, S.C. (2001) Derivation of decision rules for the evaluation of product performance using genetic algorithms and rough set theory, *Data Mining for Design and Manufacturing*, Edited by Braha, D., 337-353, Kluwer Academic Publishing.

Zhang, H., Wu, H., Ly, J. and Chen, D. (2000) Collaborative design system for performance, *Proceedings of Academia/Industry Working Conference on Research Challenges*, 59 –63, Buffalo, New York, USA.

Zhou, J. and Lin, G. (1999) Implementation of collaborative design environment based on single user CAD systems, *Third International Conference on Knowledge-Based Intelligent Information Engineering Systems*, 78-83, Adelaide, Australia.

Zhou, X., He, X., Chen, B. (2004) Method for solving computer-aided product design optimization problem based on back propagation neural network, *Chinese Journal of Chemical Engineering*, 12(4), 510-514.

Appendix 1: Case Study 1 Design Rules from Expert

- If Product Price = High AND Product Tolerance = Medium then Mold Type = Injection
- If Product Finishing = Mirror and Product Size = Small then Mold Finishing = Chrome
- If Product Tolerance = Medium then Mold Tolerance = Medium
- If Product Aspect = Normal/Squat then Mold Orientation = Any
- If Product Size = Small then Mold No of Cavities = High
- If Product Size = Large then Mold No of Cavities = Low
- If Product Hardness = High then Mold Type = Injection
- If Product Surface Finish = Mirror AND Product Size = NOT Small then Mold Finishing = Polish
- If Product Tolerance = Medium then Mold Type = Transfer
- If Gating Mark Allowed = No then Mold Type = Compression
- If Product Size = Medium then Mold No of Cavities = Low

Appendix 2: Case Study 1 Raw Data

		1		2		3		4		5		6	
Product Name		Stamped cover 1		Stamped cover 2		Stamped cover 3		Stamped cover 4		Stamped cover 5		Flex clamp seal 1	
Customer/Product Variables													
Product Price		High		High		High		High		High		High	
Product General Size		Large (>100mm)		Large (>100mm)		Small (<10mm)		Small (<10mm)		Large (>100mm)		Medium (10-100mm)	
Product Aspect Ratio		Flat/thin wall		Flat/thin wall		Flat/thin wall		Flat/thin wall		Flat/thin wall		Flat/thin wall	
Product Undercut/core		No		No		No		No		No		No	
Product Overmold		Yes		Yes		Yes		Yes		Yes		No	
Product Material Hardness		Low (40A-50A)		Low (40A-50A)		Low (40A-50A)		Low (40A-50A)		Low (40A-50A)		Low (40A-50A)	
Product Tolerance		Tight (<0.1mm)		Medium (0.1-0.25mm)		Medium (0.1-0.25mm)		Tight (<0.1mm)		Tight (<0.1mm)		Tight (<0.1mm)	
Product Surface finish		Normal		Normal		Normal		Mirror		Mirror		Normal	
Gating Mark Allowed		Yes		Yes		Yes		Yes		Yes		Yes	
Designer/Mold Variables													
Mold Type		Injection (>6 plate)		Injection (>6 plate)		Injection (>6 plate)		Injection (>6 plate)		Injection (>6 plate)		Injection (>6 plate)	
Mold Finishing		Normal		Normal		Normal		Chrome		Chrome		Normal	
Mold Insert		No		No		No		No		No		No	
Mold Orientation		Horizontal		Horizontal		Horizontal		Horizontal		Horizontal		Horizontal	
Mold Tolerance		Tight (<20um)		Medium (20-100um)		Medium (20-100um)		Tight (<20um)		Tight (<20um)		Tight (<20um)	
Mold No. of Cavities		Low		Low		High		High		Low		Low	
		7		8		9		10		11		12	
Product Name		Flex clamp seal 2		Flex clamp seal 3		Flex clamp seal 4		Crash stop 1		Crash stop 2		Crash stop 3	
Customer/Product Variables													
Product Price		High		High		Medium		Medium		Medium		Medium	
Product General Size		Medium (10-100mm)		Small (<10mm)		Medium (10-100mm)		Small (<10mm)		Small (<10mm)		Small (<10mm)	
Product Aspect Ratio		Flat/thin wall		Flat/thin wall		Flat/thin wall		Normal/squat		Normal/squat		Normal/squat	
Product Undercut/core		Yes		No		No		No		No		No	
Product Overmold		No		No		No		No		No		No	
Product Material Hardness		Low (40A-50A)		Low (40A-50A)		Low (40A-50A)		Low (40A-50A)		Low (40A-50A)		Low (40A-50A)	
Product Tolerance		Tight (<0.1mm)		Tight (<0.1mm)		Medium (0.1-0.25mm)		Tight (<0.1mm)		Tight (<0.1mm)		Tight (<0.1mm)	
Product Surface finish		Mirror		Mirror		Normal		Normal		Mirror		Textured	
Gating Mark Allowed		Yes		Yes		Yes		Yes		Yes		Yes	
Designer/Mold Variables													
Mold Type		Injection (>6 plate)		Injection (>6 plate)		Transfer (3 plate)		Transfer (3 plate)		Transfer (3 plate)		Transfer (3 plate)	
Mold Finishing		Manual polish		Chrome		Normal		Normal		Chrome		EDM	
Mold Insert		Yes		No		No		No		No		No	
Mold Orientation		Horizontal		Horizontal		Horizontal		Any		Any		Any	
Mold Tolerance		Tight (<20um)		Tight (<20um)		Medium (20-100um)		Tight (<20um)		Tight (<20um)		Tight (<20um)	
Mold No. of Cavities		Low		High		Low		High		High		High	

13		14		15		16		17		18	
Crash step 4		Crash step 5		Crash step 6		AL-SHIFA 1		AL-SHIFA 2		AL-SHIFA 3	
Customer/Product Variables											
Product Price	Medium	Medium		Low		Medium		High		High	
Product General Size	Small (<10mm)	Small (<10mm)		Small (<10mm)		Small (<10mm)		Small (<10mm)		Small (<10mm)	
Product Aspect Ratio	Normal/squat	Normal/squat		Normal/squat		Normal/squat		Normal/squat		Normal/squat	
Product Undercut/core	Yes	Yes		No		Yes		Yes		Yes	
Product Overmold	No	No		No		No		No		No	
Product Material Hardness	Low (40A-50A)	Low (40A-50A)		Low (40A-50A)		Low (40A-50A)		High (60A-70A)		High (60A-70A)	
Product Tolerance	Tight (<0.1mm)	Medium (0.1-0.25mm)		Loose (>0.25mm)		Medium (0.1-0.25mm)		Medium (0.1-0.25mm)		Medium (0.1-0.25mm)	
Product Surface finish	Normal	Textured		Normal		Normal		Normal		Textured	
Gating Mark Allowed	Yes	Yes		Yes		Yes		Yes		Yes	
Designer/Mold Variables											
Mold Type	Transfer (3 plate)	Transfer (3 plate)		Compression (2 plate)		Transfer (3 plate)		Injection (>6 plate)		Injection (>6 plate)	
Mold Finishing	Normal	EDM		Normal		Normal		Normal		EDM	
Mold Insert	Yes	Yes		No		No		Yes		Yes	
Mold Orientation	Any	Any		Any		Any		Any		Any	
Mold Tolerance	Tight (<20um)	Medium (20-100um)		Loose (>100um)		Medium (20-100um)		Medium (20-100um)		Medium (20-100um)	
Mold No. of Cavities	High	High		High		High		High		High	
19		20		21		22		23		24	
Crash step 4		Bumper 1		Bumper 2		Bumper 3		O-Ring 1		O-Ring 2	
Customer/Product Variables											
Product Price	Medium	High		Low		Low		High		High	
Product General Size	Small (<10mm)	Small (<10mm)		Small (<10mm)		Large (>100mm)		Medium (10-100mm)		Small (<10mm)	
Product Aspect Ratio	Flat/thin wall	Normal/squat		Normal/squat		Normal/squat		Normal/squat		Normal/squat	
Product Undercut/core	Yes	No		No		No		No		No	
Product Overmold	No	No		No		No		No		No	
Product Material Hardness	Low (40A-50A)	Low (40A-50A)		Low (40A-50A)		Low (40A-50A)		Low (40A-50A)		Low (40A-50A)	
Product Tolerance	Medium (0.1-0.25mm)	Tight (<0.1mm)		Medium (0.1-0.25mm)		Medium (0.1-0.25mm)		Tight (<0.1mm)		Tight (<0.1mm)	
Product Surface finish	Normal	Textured		Textured		Mirror		Mirror		Textured	
Gating Mark Allowed	Yes	Yes		No		Yes		Yes		Yes	
Designer/Mold Variables											
Mold Type	Transfer (3 plate)	Transfer (3 plate)		Compression (2 plate)		Compression (2 plate)		Transfer (3 plate)		Transfer (3 plate)	
Mold Finishing	Normal	EDM		EDM		Manual polish		Manual polish		Chrome	
Mold Insert	Yes	No		No		No		No		No	
Mold Orientation	Horizontal	Any		Any		Any		Any		Any	
Mold Tolerance	Medium (20-100um)	Tight (<20um)		Medium (20-100um)		Medium (20-100um)		Tight (<20um)		Tight (<20um)	
Mold No. of Cavities	High	High		High		Low		Low		High	

Product Name	25	26	27	28	29	30
O-Ring 3	Can filter seal 1	Can filter seal 2	Can filter seal 3	Can filter seal 4	Can filter seal 5	Can filter seal 6
Customer/Product Variables						
Product Price	Medium	Low	Low	Low	Low	Low
Product General Size	Small (<10mm)	Small (<10mm)	Small (<10mm)	Small (<10mm)	Small (<10mm)	Large (>100mm)
Product Aspect Ratio	Flat/thin wall	Flat/thin wall	Flat/thin wall	Flat/thin wall	Flat/thin wall	Normal/squat
Product Undercut/core	No	No	No	No	No	No
Product Overmold	Yes	No	No	No	No	No
Product Material Hardness	Low (40A-50A)	Low (40A-50A)	Low (40A-50A)	Low (40A-50A)	Low (40A-50A)	Low (40A-50A)
Product Tolerance	Medium (0.1-0.25mm)	Medium (0.1-0.25mm)	Medium (0.1-0.25mm)	Medium (0.1-0.25mm)	Medium (0.1-0.25mm)	Medium (0.1-0.25mm)
Product Surface finish	Normal	Normal	Normal	Loose (>0.25mm)	Mirror	Mirror
Gating Mark Allowed	Yes	Yes	No	Yes	Yes	Yes
Designer/Mold Variables						
Mold Type	Transfer (3 plate)	Compression (2 plate)	Compression (2 plate)	Compression (2 plate)	Compression (2 plate)	Compression (2 plate)
Mold Finishing	Normal	None	None	None	Chrome	Manual polish
Mold Insert	No	No	No	No	No	No
Mold Orientation	Horizontal	Horizontal	Horizontal	Horizontal	Horizontal	Any
Mold Tolerance	Medium (20-100um)	Medium (20-100um)	Medium (20-100um)	Loose (>100um)	Medium (20-100um)	Medium (20-100um)
Mold No. of Cavities	High	High	High	High	High	Low

Product Name	31	32	33
Can filter seal 6	Can filter seal 7	Can filter seal 8	Can filter seal 8
Customer/Product Variables			
Product Price	High	Medium	High
Product General Size	Large (>100mm)	Small (<10mm)	Small (<10mm)
Product Aspect Ratio	Flat/thin wall	Flat/thin wall	Flat/thin wall
Product Undercut/core	No	Yes	No
Product Overmold	No	No	No
Product Material Hardness	Low (40A-50A)	Low (40A-50A)	High (60A-70A)
Product Tolerance	Medium (0.1-0.25mm)	Medium (0.1-0.25mm)	Medium (0.1-0.25mm)
Product Surface finish	Mirror	Normal	Textured
Gating Mark Allowed	Yes	Yes	Yes
Designer/Mold Variables			
Mold Type	Injection (>6 plate)	Transfer (3 plate)	Injection (>6 plate)
Mold Finishing	Manual polish	None	EDM
Mold Insert	No	Yes	No
Mold Orientation	Horizontal	Horizontal	Horizontal
Mold Tolerance	Medium (20-100um)	Medium (20-100um)	Medium (20-100um)
Mold No. of Cavities	Low	High	High

Appendix 3: Case Study 2 Raw Data

NO	Manufacturer	Model	Dimensions	Weight	Design	Antenna	Screen Color	Screen Size	Talk Time	Standby
			mm3	gr				pixels	hr	hr
1	Handspring	Treo 270	159537	153	PDA	Stub	4096	25600	3	150
2	Handspring	Treo 300	166992	162	PDA	Stub	4096	25600	2.5	150
3	Handspring	Treo 600	147840	168	PDA	Stub	4096	25600	6	240
4	Hitachi	G1000	284004	238	PDA	Stub	65000	76800	2.5	168
5	Hitachi	SH-P300	125840	103	Block	Extend	4096	19440	3.5	240
6	Kyocera	7135	187860	187	PDA	Extend	65000	2560	3.5	160
7	LG	C1300	73143	81	Clam	Stub	65000	16384	3.4	240
8	LG	G4050	102856	91	Clam	Extend	65000	20480	3	200
9	LG	L1400	106800	95	Clam	Stub	65000	20480	3.3	220
10	LG	LX5550	102528	110	Clam	Extend	65000	19200	2.5	144
11	LG	VI-5225	106800	99	Clam	Internal	4096	19200	2.5	192
12	LG	VX8000	110400	110	Clam	Extend	262000	38720	3.33	110
13	Motorola	A845	191800	155	Block	Internal	65000	38720	2	215
14	Motorola	C650	90640	90	Block	Internal	65000	16384	6.33	215
15	Motorola	MPx200	115344	118	PDA	Internal	65000	38720	4.25	104
16	Motorola	RAZR V3	72716	95	Clam	Internal	262000	38720	3	200
17	Motorola	T720	87843	101	Clam	Stub	4096	19200	6.5	230
18	Motorola	T730	97290	115	Clam	Extend	4096	19200	2.75	195
19	Motorola	V220	88704	100	Clam	Extend	65000	16384	5.33	230
20	Motorola	V810	79464	85	Clam	Extend	260000	20480	3.75	285
21	Nokia	3100	65790	85	Block	Internal	4096	16384	6	410
22	Nokia	3660	192660	130	Block	Internal	65000	36608	4	200
23	Nokia	5100	118156.5	104	Block	Internal	4096	16384	5	240
24	Nokia	6820	100580	100	Clam	Internal	4096	16384	7	240
25	Nokia	N-Gage	187600	137	Side	Internal	4096	36608	3	175
26	Panasonic	GD87	110446	103	Clam	Stub	65000	23232	5	150
27	Panasonic	GU87	110446	103	Clam	Stub	65000	23232	5	150
28	Panasonic	X70	98136	92	Clam	Stub	65000	23232	5	250
29	Samsung	MM-A700	111250	119	Clam	Extend	262000	38720	4	216
30	Samsung	SCH-I600	111780	130	Clam	Stub	65000	38720	4.17	240
31	Samsung	SGH-X427	86480	79	Clam	Stub	65000	20480	5	200
32	Samsung	SPH-I700	147840	162	PDA	Stub	65000	76800	2.2	130
33	Siemens	A60	118910	85	Block	Internal	4096	8080	5	250
34	Siemens	C62	88880	85	Block	Internal	4096	16384	4.5	250
35	Siemens	M56	97566	83	Block	Internal	4096	8080	5	250
36	Siemens	SL56	79376	79	Slide	Internal	4096	8080	3.5	175
37	Siemens	ST60	106722	87	Block	Internal	65000	19200	4.5	140
38	Siemens	SX56	169506	196	PDA	Stub	4096	76800	3.5	150
39	Sony Ericsson	K700	93060	93	Block	Internal	65000	38720	7	360
40	Sony Ericsson	P900	157320	150	PDA	Internal	65000	66560	16	480
41	Sony Ericsson	S700	132300	137	Swivel	Internal	262000	76800	7	300
42	Sony Ericsson	T316	101920	97	Block	Internal	256	8080	7.5	350
43	Sony Ericsson	T68i	96000	84	Block	Internal	256	8080	7.5	384
44	Sony Ericsson	Z600	116640	106	Clam	Internal	65000	20480	6	200

NO	Manufacturer	Model	Camera	FM Radio	Loudspeaker	MP3 Player	Bluetooth	GPRS
			pixels					
1	Handspring	Treo 270	UA	No	Yes	No	No	Yes
2	Handspring	Treo 300	UA	No	Yes	No	No	Yes
3	Handspring	Treo 600	307200	No	Yes	Yes	No	Yes
4	Hitachi	G1000	307200	No	Yes	Yes	No	Yes
5	Hitachi	SH-P300	UA	No	Yes	No	No	Yes
6	Kyocera	7135	UA	No	Yes	Yes	No	Yes
7	LG	C1300	UA	No	No	No	No	Yes
8	LG	G4050	Add on	No	No	No	No	Yes
9	LG	L1400	307200	No	No	No	No	Yes
10	LG	LX5550	UA	No	Yes	No	No	Yes
11	LG	VI-5225	UA	No	Yes	No	No	Yes
12	LG	VX8000	1363148.8	No	Yes	Yes	No	Yes
13	Motorola	A845	Yes	No	Yes	Yes	Yes	Yes
14	Motorola	C650	307200	No	Yes	No	No	Yes
15	Motorola	MPx200	UA	No	Yes	Yes	No	Yes
16	Motorola	RAZR V3	307200	No	Yes	No	Yes	Yes
17	Motorola	T720	UA	No	Addon	Addon	No	Yes
18	Motorola	T730	UA	Addon	Addon	Addon	No	Yes
19	Motorola	V220	307200	Addon	Yes	No	No	Yes
20	Motorola	V810	307200	Yes	Yes	No	No	Yes
21	Nokia	3100	Add on	Addon	Yes	No	No	No
22	Nokia	3660	307200	No	Yes	No	Yes	Yes
23	Nokia	5100	Add on	Yes	Yes	No	No	Yes
24	Nokia	6820	101376	Addon	Yes	No	Yes	Yes
25	Nokia	N-Gage	UA	Yes	Yes	Yes	Yes	Yes
26	Panasonic	GD87	23232	No	No	No	No	Yes
27	Panasonic	GU87	23232	No	No	No	No	Yes
28	Panasonic	X70	23232	No	No	No	Yes	Yes
29	Samsung	MM-A700	995328	Addon	No	No	No	Yes
30	Samsung	SCH-I600	UA	No	Yes	No	No	Yes
31	Samsung	SGH-X427	UA	No	No	No	No	Yes
32	Samsung	SPH-I700	307200	No	Yes	Yes	No	Yes
33	Siemens	A60	UA	No	No	No	No	Yes
34	Siemens	C62	Add on	No	Yes	No	No	Yes
35	Siemens	M56	Add on	No	Yes	No	No	Yes
36	Siemens	SL56	Add on	No	Yes	No	No	Yes
37	Siemens	ST60	307200	No	Yes	No	No	Yes
38	Siemens	SX56	UA	No	Yes	Yes	No	Yes
39	Sony Ericsson	K700	307200	Yes	Yes	Yes	Yes	Yes
40	Sony Ericsson	P900	307200	Addon	Yes	Yes	Yes	Yes
41	Sony Ericsson	S700	1228800	Yes	Yes	Yes	Yes	Yes
42	Sony Ericsson	T316	Add on	Addon	No	No	No	Yes
43	Sony Ericsson	T68i	Add on	Addon	No	Addon	Yes	Yes
44	Sony Ericsson	Z600	101376	Addon	No	Addon	Yes	Yes

NO	Manufacturer	Model	No Frill	Fancy	Pro	Tech
1	Handspring	Treo 270	Acceptable	Unacceptable	Unacceptable	Unacceptable
2	Handspring	Treo 300	Acceptable	Unacceptable	Unacceptable	Unacceptable
3	Handspring	Treo 600	Excessive	Acceptable	Excessive	Unacceptable
4	Hitachi	G1000	Excessive	Acceptable	Excessive	Unacceptable
5	Hitachi	SH-P300	Acceptable	Unacceptable	Unacceptable	Unacceptable
6	Kyocera	7135	Excessive	Acceptable	Excessive	Unacceptable
7	LG	C1300	Acceptable	Acceptable	Acceptable	Unacceptable
8	LG	G4050	Excessive	Acceptable	Acceptable	Unacceptable
9	LG	L1400	Excessive	Acceptable	Acceptable	Unacceptable
10	LG	LX5550	Acceptable	Unacceptable	Unacceptable	Unacceptable
11	LG	VI-5225	Acceptable	Unacceptable	Unacceptable	Unacceptable
12	LG	VX8000	Excessive	Excessive	Excessive	Acceptable
13	Motorola	A845	Excessive	Acceptable	Excessive	Unacceptable
14	Motorola	C650	Excessive	Unacceptable	Unacceptable	Unacceptable
15	Motorola	MPx200	Acceptable	Acceptable	Excessive	Acceptable
16	Motorola	RAZR V3	Excessive	Acceptable	Acceptable	Acceptable
17	Motorola	T720	Excessive	Unacceptable	Acceptable	Unacceptable
18	Motorola	T730	Excessive	Unacceptable	Unacceptable	Unacceptable
19	Motorola	V220	Excessive	Acceptable	Acceptable	Unacceptable
20	Motorola	V810	Excessive	Acceptable	Excessive	Unacceptable
21	Nokia	3100	Excessive	Unacceptable	Unacceptable	Unacceptable
22	Nokia	3660	Excessive	Unacceptable	Unacceptable	Acceptable
23	Nokia	5100	Excessive	Unacceptable	Excessive	Unacceptable
24	Nokia	6820	Excessive	Acceptable	Acceptable	Acceptable
25	Nokia	N-Gage	Excessive	Acceptable	Excessive	Acceptable
26	Panasonic	GD87	Excessive	Acceptable	Acceptable	Unacceptable
27	Panasonic	GU87	Excessive	Acceptable	Acceptable	Unacceptable
28	Panasonic	X70	Excessive	Acceptable	Acceptable	Acceptable
29	Samsung	MM-A700	Excessive	Excessive	Acceptable	Unacceptable
30	Samsung	SCH-I600	Acceptable	Unacceptable	Acceptable	Unacceptable
31	Samsung	SGH-X427	Acceptable	Acceptable	Acceptable	Unacceptable
32	Samsung	SPH-I700	Excessive	Acceptable	Excessive	Unacceptable
33	Siemens	A60	Acceptable	Unacceptable	Acceptable	Unacceptable
34	Siemens	C62	Excessive	Unacceptable	Acceptable	Unacceptable
35	Siemens	M56	Excessive	Unacceptable	Acceptable	Unacceptable
36	Siemens	SL56	Excessive	Acceptable	Acceptable	Unacceptable
37	Siemens	ST60	Excessive	Unacceptable	Unacceptable	Unacceptable
38	Siemens	SX56	Excessive	Acceptable	Unacceptable	Unacceptable
39	Sony Ericsson	K700	Excessive	Acceptable	Excessive	Acceptable
40	Sony Ericsson	P900	Excessive	Acceptable	Unacceptable	Unacceptable
41	Sony Ericsson	S700	Excessive	Excessive	Excessive	Acceptable
42	Sony Ericsson	T316	Excessive	Unacceptable	Acceptable	Unacceptable
43	Sony Ericsson	T68i	Excessive	Acceptable	Acceptable	Acceptable
44	Sony Ericsson	Z600	Excessive	Excessive	Acceptable	Acceptable