

Asking the Right Questions to Elicit Product Requirements

Min Wang

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Applied Science
(Electrical and Computer Engineering) at
Concordia University
Montreal, Quebec, Canada

August 2007

© Min Wang, 2007



Library and
Archives Canada

Bibliothèque et
Archives Canada

Published Heritage
Branch

Direction du
Patrimoine de l'édition

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 978-0-494-34785-0

Our file Notre référence

ISBN: 978-0-494-34785-0

NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protègent cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.


Canada

ABSTRACT

Asking the Right Questions to Elicit Product Requirements

Min Wang

Eliciting precise and comprehensive product requirements from customers is of critical importance for the success of product development. In this thesis, a generic process is proposed for eliciting product requirements by asking questions generated by a linguistic analysis. The linguistic analysis is based on a graphic language called Recursive Object Model (ROM). Two types of questions are asked in the process. One is independent of the domains of product development whereas the other is based on the product domain. A generic template is developed for generating the questions and for determining the sequence of asking those questions. The answers to the questions can be sought from the customers and other partners involved in the product development, designer's own knowledge and experience, text books, dictionary, the internet, and/or the nature itself. A software prototype is developed to support the proposed process. Two case studies are used to illustrate the question generation process.

Acknowledgements

The first person that should be acknowledged for his roles in the present thesis is my supervisor, Dr. Yong Zeng. I owe an immense debt of gratitude to him for his knowledgeable and helpful suggestions and for his constant encouragement and support. Dr. Zeng has offered me extensive and thoughtful comments throughout my entire master's program. The comments and the patience that he has displayed are very much appreciated. I have benefited enormously from the discussions with Dr. Zeng, especially from his inspiring guidance in research methods, which will prove to be helpful in my future study and work. The outstanding organizational skills that Dr. Zeng has displayed have made a friendly, efficient and cooperative working environment for the Design Group in CIISE.

My gratitude also goes to Lei Chen, Baiquan Yan, Shenji Yao and other members in the group, who have shared all the joyful and bitter moments with me in the past two years. Their invaluable suggestions and friendship is an asset in my life. A heartfelt note of thanks goes to the professors and staffs in CIISE, who have provided a wonderful academic surroundings for me to complete this study.

My husband has been especially supportive of my efforts to pursue this study, exhibiting his extraordinary understanding and encouragement.

Table of Contents

List of Figures	vii
List of Tables.....	ix
Chapter 1 Introduction	1
1.1 Motivation	1
1.2 Objective	3
1.3 Challenges	4
1.4 Contributions.....	4
1.5 Thesis organization	5
Chapter 2 Literature Review	6
2.1 Design, design problem and design process.....	6
2.2 Product requirements.....	9
2.3 Requirements elicitation.....	10
Chapter 3 Theoretical Foundation: Environment Based Design.....	15
3.1 Environment based design (EBD): brief introduction.....	15
3.2 Logic of design.....	18
3.3 Design state: mathematical foundation for representing design.....	20
3.4 Recursive object model (ROM)	22
Chapter 4 Asking the Right Questions to Elicit Product Requirements.....	24
4.1 General steps for product requirements elicitation.....	24
4.2 Asking the generic questions.....	31
4.3 Asking the domain questions	36
4.4 Case study of EEG experiment design	42
Chapter 5 Software Prototype for Eliciting Product Requirements	52
5.1 Problem formulation	52
5.2 System architecture	53
5.3 System interfaces.....	56
5.4 Case study	63
Chapter 6 Conclusions and Future Work	71
6.1 Overview	71
6.2 Conclusions	71

6.3 Future work	72
Publications	74
References	75

List of Figures

Figure 1 Illustration of problems existing in software product development adapted from (Unknown-Author, 2007).....	1
Figure 2 Ullman's generic design process in all projects (Ullman, 2002)	8
Figure 3 Environment based design: process flow.....	16
Figure 4 The evolution process of design (Zeng, 2004b).....	17
Figure 5 State space of design under synthesis and evaluation operators (Zeng, 2004b)	18
Figure 6 Evolution of the design process (Zeng and Gu, 1999b).....	19
Figure 7 Product system.....	21
Figure 8 Generic inquiry process for requirements elicitation.....	25
Figure 9 Asking the generic questions	31
Figure 10 ROM diagram for EEG experiment design.....	34
Figure 11 Seven events in product life cycle (Chen and Zeng, 2006).....	37
Figure 12 Waterfall model in software engineering (Schach, 2002).....	38
Figure 13 Eight levels of requirements (Chen and Zeng, 2006)	40
Figure 14 Ask domain specific questions.....	42
Figure 15 ROM diagram for Step 1.....	43
Figure 16 ROM diagram for A1	45
Figure 17 ROM diagram for A2.....	46
Figure 18 ROM diagram for A4.....	47
Figure 19 ROM diagram for A5.....	47
Figure 20 Merged ROM diagram for Step 4	48
Figure 21 Part of the final ROM diagram	51
Figure 22 System architecture of the software prototype.....	54
Figure 23 Module of design problem analysis	55
Figure 24 Module of design problem extension.....	56
Figure 25 Interface for login and authentication	57
Figure 26 Interface for homepage	58
Figure 27 Interface for design problem input.....	59
Figure 28 Interface for ROM drawing	60
Figure 29 Interface for question generation	61
Figure 30 Interface for question listing	62

Figure 31 Internal drum brake (Hubka et al., 1988).....	63
Figure 32 ROM diagram for Step 1.....	64
Figure 33 ROM diagram for Step 4.....	65
Figure 34 ROM diagram for A4.....	67
Figure 35 ROM diagram for A5.....	68
Figure 36 ROM diagram for A6.....	68
Figure 37 ROM diagram for A7.....	68
Figure 38 ROM diagram for A8.....	68
Figure 39 ROM diagram for A10.....	69
Figure 40 ROM diagram for A11.....	69
Figure 41 ROM diagram for Step 7.....	69
Figure 42 Final ROM diagram.....	70

List of Tables

Table 1 Symbols in ROM.....	23
Table 2 Rules for objects analysis.....	32
Table 3 Questions template for object analysis.....	35
Table 4 Classification of nouns	35
Table 5 Questions generation rules for Step 5.....	41
Table 6 Questions for Step 2	44
Table 7 Answers for Step 3	45
Table 8 Questions for Step 7	49
Table 9 Questions for Step 8	49
Table 10 Questions for Step 2	64
Table 11 Answers for Step 3	65
Table 12 Questions for Step 5	66
Table 13 Answers for Step 6	67

Chapter 1

Introduction

1.1 Motivation

An interesting cartoon has been distributed in the community of software engineering and quality engineering, a part of which is shown in Figure 1 (Unknown-Author, 2007). This cartoon introduces typical problems existing in the software product development process.

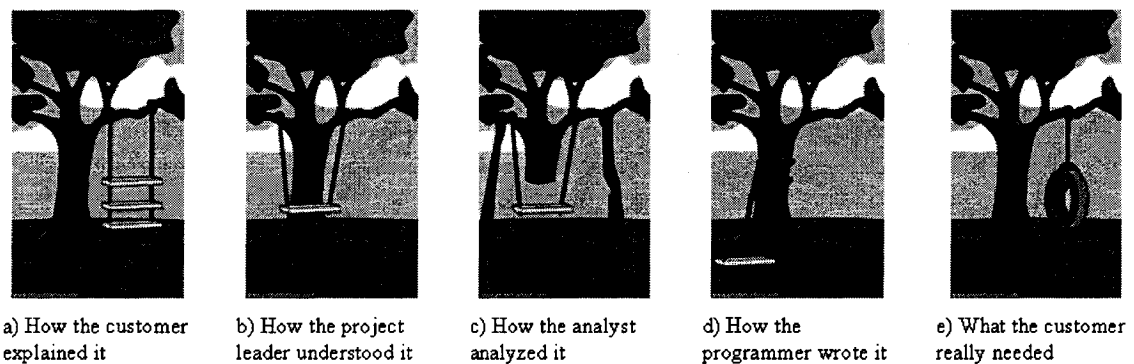


Figure 1 Illustration of problems existing in software product development adapted from (Unknown-Author, 2007)

A few observations, among others, can be made from the cartoon above:

- 1) Product requirements and design descriptions may be misinterpreted due to the ambiguity of natural language in the process of communication from the customer to project manager, to system analyst, to programmer, etc., which is demonstrated by the transformation such as that from a) to b) in Figure 1. To reduce the misunderstanding of product requirements, a language is needed to easily transform the correct

meanings of the product requirements to all the partners in the product development process.

- 2) The customers may not know exactly what they want, which is shown by the gap between a) and e) in Figure 1. It is essential to find out the customer's real intent during the product development process.
- 3) Implicit product requirements may be ignored in the process of product development, as is shown in Figure 1 c), where the requirement is implicit in that if the trunk of a tree is cut then the tree is going to die. The product development process should also constitute the identification of complete product requirements.

The three problems presented above exist not only in the area of software engineering but also in manufacturing engineering and almost all domains involved with human beings. In product development, those problems often result in the poor specification of requirements at the early stages. The propagation of those problems into the later stages of product development will lead to a failed project or will lead to problems that are costly to be fixed.

Therefore, it is critical to study methodologies that will enhance the communications between various partners during a product development process, that will identify the customer's real intent, and that will gather a relatively complete list of requirements.

1.2 Objective

To deal with the first problem introduced in Section 1.1, Dr. Yong Zeng proposed a graphic language, Recursive Object Model (ROM) (Zeng, 2007), for the effective communication of design ideas. The objectives of this thesis are as follows:

- 1) identify the customer's real intent
- 2) collect the complete list of product requirements.

Product requirements are descriptions of desired solutions to a design problem. The requirements can be described in a natural language, sketches, equations, and some other forms like multimedia. Since customers usually describe their intent with natural language, the scope of the research presented in this thesis is limited in the product requirements described in natural language. However, a natural language may easily lead to an ambiguous or distorted understanding of the user's original intents (Oxman, 2004). This thesis aims to identify the customer's real intent through clarifying the meaning of the requirements by asking the right questions.

In a product development process, some commonly recognized requirements are often taken for granted and may not be properly communicated between partners. As a result, those requirements could be ignored, which will lead to faulty products. Based on the classification of product requirements, which defines the scope of product requirements (Chen and Zeng, 2006; Zeng, 2004b), questions will be generated to lead the designers to complete the specification of relevant requirements.

Therefore, a systematic procedure will be proposed in this thesis for the generation of right questions to elicit relatively precise and complete product requirements.

In order to achieve the objective above, the following are the specific aims of this thesis:

- 1) to generate the right questions based on linguistic analysis of natural language text describing product requirements,
- 2) to develop domain dependent rules that will lead to meaningful questions, and
- 3) to develop a software prototype that supports the generation of questions for requirements elicitation.

1.3 Challenges

While for over 150 years mathematicians have studied the logic of statement, the mathematics of questions has been almost entirely neglected (Knuth, 2005). Question asking has remained mostly an experience based practice. The difficulty in question asking lies in the capturing of semantics implied in an interested text and the identification of missing information.

1.4 Contributions

Based on the recursive object model (ROM) and classification of product requirements (Chen and Zeng, 2006), this thesis has made two major contributions:

- 1) An algorithm for asking generic questions for clarifying the meaning of customer's requirements

- 2) An algorithm for asking domain specific questions for gathering complete product requirements relevant to the domain.

A software prototype is developed to implement the algorithms. The answering of those questions is not the concern of this present thesis. We simply collect answers from all available resources manually and feed those answers back into the question asking system. Two case studies are provided to demonstrate how the proposed algorithms work.

1.5 Thesis organization

Chapter 1 introduces the present thesis, presents the motivation, significance, objectives and overview of this thesis.

Chapter 2 examines the previous research dealing with the requirements elicitation.

Chapter 3 provides the theoretical foundation of this thesis, including the concepts of design thinking, design state, environment based design and recursive object model.

Chapter 4 presents a generic iterative inquiry approach for eliciting precise and complete product requirements, which is demonstrated by a case study of an experiment design. To support the application of the proposed approach, a software prototype system is presented in Chapter 5. An example of rivet tool design is chosen to illustrate the approach and the software prototype system.

Chapter 6 summarizes the main research results from the present thesis and points out some future research directions.

Chapter 2

Literature Review

The objective of the present thesis is to develop methods for understanding customer's real intent and for identifying implicit product requirements, which will enhance the quality of design. To achieve this objective, this literature review will cover the following areas:

- Design, design problem and design process, which provides the context for the present research
- Product requirements and its definition and classification
- Requirements elicitation methods.

2.1 Design, design problem and design process

Researchers have provided various descriptions of the term “design” such as: design activities are generally considered to be a form of complex problem solving (Simon, 1969); design begins with a needs-analysis (Asimow, 1962); design is a social activity (Minneman, 1991). In some design studies, the objectives usually focus on finding common characteristics from different engineering domains, within the framework of cognitive science (Prabhakar and Goel, 1998). Therefore, design process can be regarded as a cognitive process intended to produce a solution to a design task.

Design problems are well-known as ill-defined and open-ended problems. An ill-defined problem, also called an ill-structured problem as distinguished from a well-structured problem, involves three critical concepts in the cognitive science. Different from a well-defined problem, such as a chess game, an ill-defined problem (a) is more complex, (b) begins with the inadequate initial conditions, and (c) presents fewer “end” criteria. An open-ended problem does not have an optimal solution, but only a satisfying one (Simon, 1969), which may have several or many correct solutions. When provided with the same list of product requirements, different design teams may produce different product solutions.

The design process varies from product to product and from industry to industry. A generic diagram of the activities that must be accomplished for all projects, is shown in Figure 2 (Ullman, 2002). In this framework, any product must go through five phases: project definition, specification definition, conceptual design, the product development and product support.

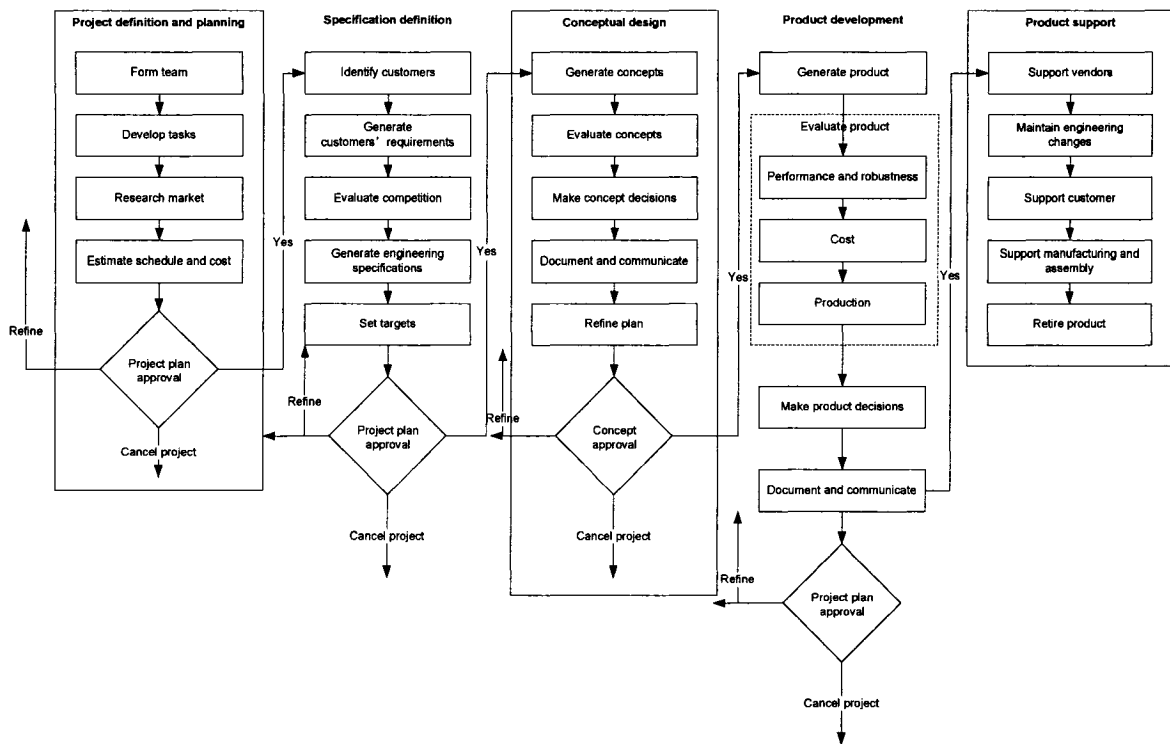


Figure 2 Ullman's generic design process in all projects (Ullman, 2002)

The nature of design requirements and the design process have been the subject of a wide variety of research. Recently, some approaches have been proposed in this field, such as methodology-based design and language-based design (Darlington and Culley, 2002). In the category of methodology-based design, some methods for the development of design support mechanisms have been applied, such as quality function deployment (QFD) (Akao and Glenn, 2003; Clausing, 1998), a taxonomic approach (Gershenson and Stauffer, 1999), key characteristics (Lee and Thornton, 1996), and functional decomposition. Darlington and Culley classified the research in this category into two kinds of noticeable design theories (Darlington and Culley, 2002). One comes from Wootton who made an analysis of the design requirement process in terms of the

stakeholders and information sources involved in the complexity of developing new products as corporate activity (Wootton et al., 1998). The theory provides the foundation of a prescriptive guide to the process of requirement capturing for industrial use. The other one, a science-based approach to product design theory, comes from Zeng and Gu (Zeng and Gu, 1999a; Zeng and Gu, 1999b). They proposed a set theory-based representation scheme for the representation of the design objects that evolve during the design process. As the continuation of the efforts in the science-based approach to product design theory, Zeng proposed a new design methodology, environment based design (EBD), which is a step-by-step approach to solving a poorly defined problem and which can assist the designers in delivering creative and innovative design solutions (Zeng, 2004a; Zeng, 2004b).

2.2 Product requirements

“Requirement” has the following definitions by IEEE 90 (Unknown-Author, 1983):

- (1) A condition or capability needed by a user to solve a problem or to achieve an objective;
- (2) A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents;
- (3) A documented representation of a condition or capability as in (1) or (2).

Southwell et al. (Southwell et al., 1987) classified requirements into: functional and nonfunctional requirements are classified further as performance/reliability, interfaces and design constraints.

The requirements modeling process is divided into four steps: requirements elicitation, requirements analysis, requirements management and requirements verification (Nuseibeh and Easterbrook, 2000; Unknown-Author, 1983).

The IEEE Guide to Software Requirements Specifications defines a good software requirements specification as being: unambiguous, complete, verifiable, consistent, modifiable, traceable, usable, and necessary. A good requirements elicitation process must support the development of a specification with these attributes. A specification describes a system and its desired properties. The process of specification is the act of writing things down precisely. The formal specification uses a language with a mathematically defined syntax and semantics (Clarke and Wing, 1996). According to Clarke and Wing's perspective (Clarke and Wing, 1996), formal methods should be described as mathematically based languages, techniques and tools for specifying and verifying safety-critical systems.

2.3 Requirements elicitation

Requirements elicitation is a process of interactions between customers, designers, project managers, and other partners of the product development. Natural language is usually a major means of communication during the elicitation process (Lecoeuche et al.,

1998). Natural language allows design requirements to be discussed with enormous semantic richness easily and naturally by non-specialists. However, natural language descriptions carry lots of noises, ambiguities, and contradictions, as pointed out by Meyer (Meyer, 1985). Therefore, requirements elicitation has to deal with informality, incompleteness and inconsistency (Leite and Cesar, 1987). Researchers, including Christel and Kang, Hickey and Davis (Hickey and Davis, 2004), summarized some major methods in requirements elicitation as follows:

- Interviews: interviews are the most common technique used for gathering information during requirements elicitation. However there are no standardized procedures for structuring information received from interviews (Zeroual, 1989). It is also challenging to integrate different interpretations, goals, objectives, communication styles, and use of terminology into a single set of requirements.
- Issue-based information system (IBIS) (Christel and Kang, 1992; Conklin *et al.*, 1991): IBIS provides an integrated approach to organizing information from interviews, though it does not support automated checking of consistency, nor support for types outside of issues, positions, and arguments.
- Joint application design (JAD): JAD, a team technique, focuses on improving the group process and getting the right people involved from the beginning (Zahniser, 1990). It promotes the cooperation, understanding, and teamwork. Meanwhile, JAD enhances idea generation and evaluation, communication, and consensus generation. JAD is specifically designed for the development of large computer systems and it has been used successfully by IBM since the late 1970s (Wood and Silver, 1995).

- Misuse cases: Misuse cases apply the concept of a negative scenario in a use-case context. One significant characteristic of misuse cases is that they seem to lead to quality requirements, such as those for safety and security, whereas other elicitation methods are focused on end-user requirements (Alexander, 2003).

In order to promote understanding and the gathering of information in elicitation, many current elicitation approaches represent the requirements from different viewpoints such as

- Controlled requirements expression (CORE): CORE provides a framework for analyzing and expressing requirements in a structured diagrammatic notation (Christel and Kang, 1992; Mullery, 1979). However, CORE does not effectively represent timing behaviour and reuse; the support of complex data descriptions remains to be a problem.
- Feature-oriented domain analysis (FODA) (Kang et al., 1990): FODA is a domain analysis method that focuses on developing reusable assets. The FODA method was founded on two modeling concepts: abstraction and refinement (Kean, 1997). Abstraction is used to create domain products from the specific applications in the domain. These generic domain products abstract the functionality. The generic nature of the domain products is created by abstracting factors that make one application different from other related applications. The FODA method abstracts different applications to the level where no differences exist between the applications. Specific applications in the domain are developed as refinements of the domain products.

- Critical discourse analysis (CDA) (Schiffrin, 1994): CDA uses sociolinguistic methods to analyze verbal and written discourse. Sociolinguistics assigns special significance to the structure of speech and texts; it also provides methods for specifying the linguistic features of different types of discourse units and the way they are tied together into larger units of meaning (Alvarez, 2002). In particular, CDA can be used to analyze interviews from requirements elicitation and to understand the narratives and "stories" that emerge during the interviews.
- Accelerated Requirements Method (ARM) (Hubbard et al., 2000): The ARM process is a facilitated requirements elicitation and description activity. Overall, there are three phases of the process: preparation phase, facilitated session phase and deliverable closure phase. During the preparation phase, planning and preparation are completed to ensure an effective session. During the session phase, a trained--and content neutral--facilitator leads the selected participants through a structured process to collect the functional requirements of the project under consideration. And in the closure phase, the key deliverables, such as a requirements collection, are polished.
- Quality Function Deployment QFD (QFD-Institute, 2005): QFD is "an overall concept that provides a means of translating customer requirements into the appropriate technical requirements for each stage of product development and production (QFD-Institute, 2005). The distinguishing attribute of QFD is the focus on customer needs throughout all product development activities. By using QFD, organizations can promote teamwork, prioritize action items, define clear objectives, and reduce development time (QFD-Institute, 2005). Although QFD covers a broad portion of the

product development life cycle, the earlier stages of the QFD process are applicable to requirements elicitation for software engineering (Mead, 2006). These stages include: 1) identifying the customer (stakeholders), 2) gathering high-level customer requirements, 3) constructing a set of system features that can satisfy customer needs, and 4) creating a matrix to evaluate system features against satisfaction of customer needs.

Though a lot of efforts have been made to address the problems in requirements elicitation, not much research results have been reported regarding the approaches based on questioning and answering. A recent investigation was Eris's work on the role of effective inquiry in the innovative engineering design process (Eris, 2004). But his work falls short of a methodology on how to make effective inquiries.

Chapter 3

Theoretical Foundation: Environment Based Design

The work introduced in this thesis serves a new design methodology: environment based design (EBD) proposed by Dr. Zeng (Zeng, 2002; Zeng, 2003; Zeng, 2004a; Zeng, 2004b; Zeng and Cheng, 1991; Zeng and Gu, 1999a; Zeng and Gu, 1999b; Zeng *et al.*, 2004). Environment based design is a few things. Firstly, it is a prescriptive model of design that guides designers from the elicitation of customer requirements throughout the generation and evaluation of design concepts. Secondly, it is a descriptive model of the design process that illustrates how designers accomplish a design task. Thirdly, it is a logical result derived from the axiomatic theory of design modeling (Zeng, 2002). To facilitate the presentation of the research results introduced in this thesis, basic concepts in the environment based design will be reviewed in this section.

3.1 Environment based design (EBD): brief introduction

Different from traditional design methodologies, which are largely based on the understanding that a generic design process comprises analysis, synthesis, and evaluation, the environment based design methodology includes the following three main stages: environment analysis, conflict identification, and concept generation. These three stages work together to generate progressively and simultaneously and refine the design specifications and design solutions. The EBD methodology is illustrated in Figure 3.

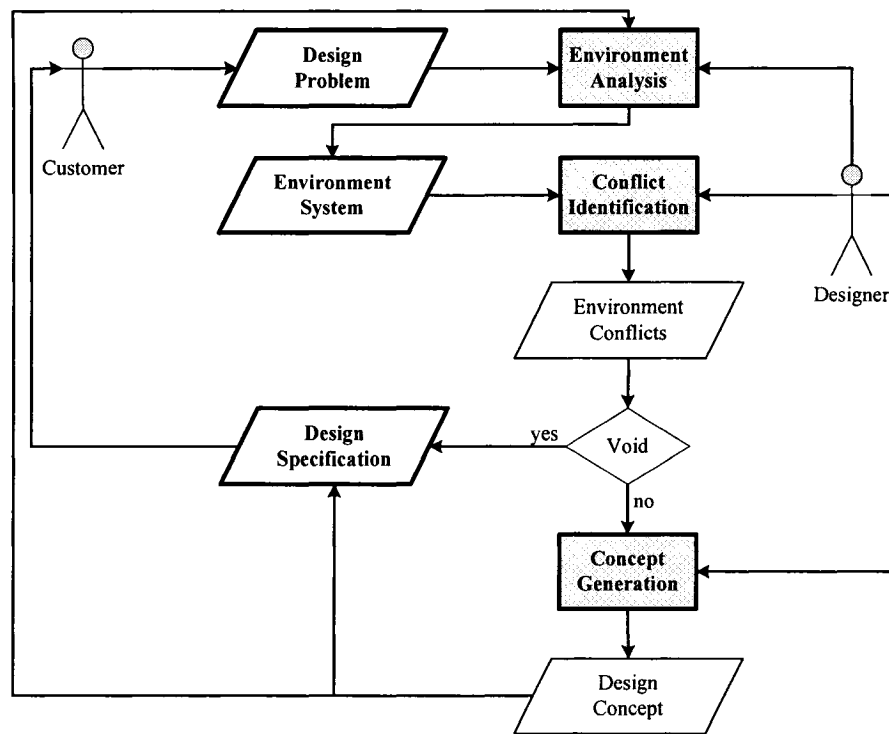


Figure 3 Environment based design: process flow

The objective of environment analysis is to find out the key environment components, in which the product works, and the relationships between the environment components. From the environment implied in the design problem described by the customer(s), the designer will introduce extra environment components that are relevant to the design problem at hands. The results from this analysis constitute an environment system. One of the key methods for environment analysis is linguistic analysis (Chen *et al.*, 2007). Following the environment analysis, conflicts should be identified among the relations between environment components. At the third stage of EBD, a set of key environment conflicts will be chosen to be resolved by generating some design concepts. This process continues until no more unacceptable environment conflicts exist.

The EBD methodology is based on the following research results:

- Design is a recursive process to generate design solutions that can pass the evaluation of a set of criteria that are defined by the design solutions to be evaluated (Zeng and Cheng, 1991).
- The recursive structure of design can be formally represented by the evolution of both design requirements and product descriptions (Zeng and Gu, 1999a; Zeng and Gu, 1999b).
- The state of design evolution, which consists of design requirements and product descriptions, can be represented by a structure operation \oplus and an interaction operation \otimes on an object \mathbf{O} (Zeng, 2002):

$$\oplus E = E \cup (E \otimes E) \quad (3.1)$$

where E is the product environment. Figure 4 shows the evolution of the design states throughout the environment based design process.

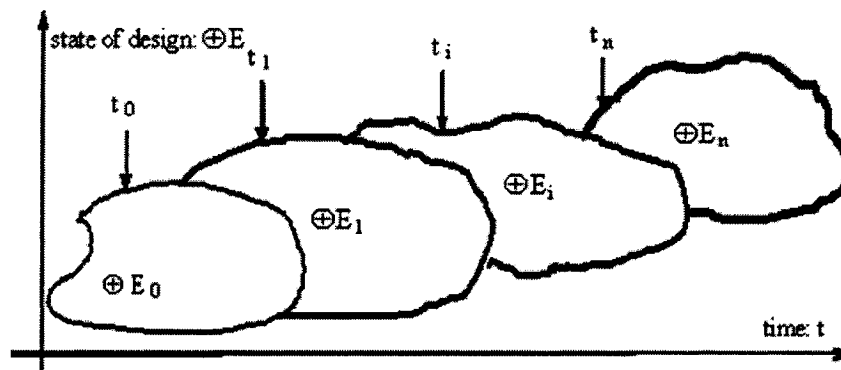


Figure 4 The evolution process of design (Zeng, 2004b)

- The source of design requirements is product environment E (Zeng, 2004b). During the environment based design process, the evolution from the design state $\oplus E_i$ to the design state $\oplus E_{i+1}$ is governed by the following design governing equation (Zeng, 2004b), where K_i^s and K_i^e are evaluation and synthesis operators, respectively.

$$\oplus E_{i+1} = K_i^s(K_i^e(\oplus E_i)). \quad (3.2)$$

The synthesis operator stretches the state space of design whereas the evaluation operator folds and reduces the state space. The final design solution is the balance of those two forces. This governing equation is indeed another form of the recursive logic of design (Zeng and Cheng, 1991). Figure 5 illustrates this governing equation.

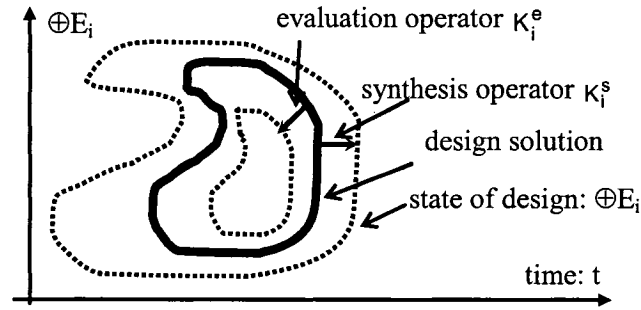


Figure 5 State space of design under synthesis and evaluation operators (Zeng, 2004b)

The results summarized above will be discussed in more details in the next sections.

3.2 Logic of design

The traditional view of the design process is that design evolution goes through the following stages: specification of design requirements, design synthesis, and design evaluation. These three stages iterate until a satisfying design solution is found.

Zeng and Cheng (Zeng and Cheng, 1991) indicated that design is a recursive process in which a satisfying design solution must pass an evaluation defined by the design knowledge that is recursively dependent on the design solution to be evaluated. Since the design knowledge, which implies the design criteria, is part of the design problem, the generation of design solutions indeed changes the original design problem. This observation leads to the proposal of the recursive logic as the logic of design (Zeng and Cheng, 1991). Based on this logic, the design process is described as a series of design states defined by both product descriptions and product requirements, as is shown in Figure 6 (Zeng and Gu, 1999b), where design requirements and design solutions co-evolve throughout the design process. Therefore, it is fundamentally impossible to distinguish design problem and design solutions.

When the word design problem is used, the design problem is given with a partial solution attached. When the word design solution is used, the design solution is bundled with further design problems. In this thesis, design states, design problem, and design solutions will be used interchangeably to mean the state of design.

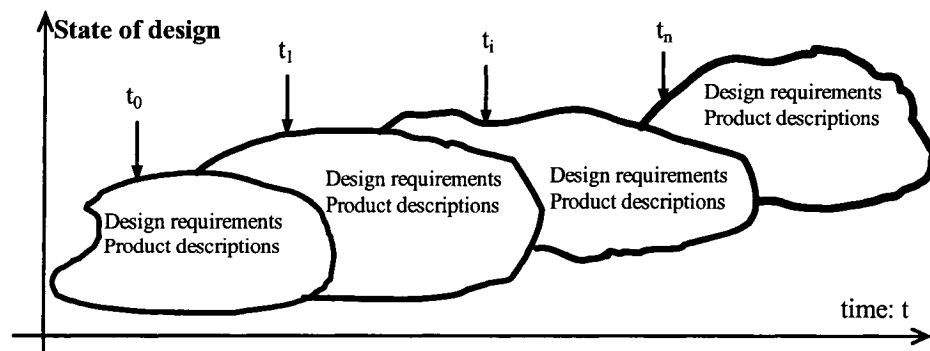


Figure 6 Evolution of the design process (Zeng and Gu, 1999b)

3.3 Design state: mathematical foundation for representing design

To represent the state of design given in Figure 6, Zeng proposed a new mathematical tool, named axiomatic theory of design modeling (Zeng, 2002), which can be used to represent both design solutions (Zeng et al., 2004) and design requirements (Chen and Zeng, 2006; Zeng, 2004b). To facilitate its application to engineering problems, a graphic language, recursive object model (ROM) (Zeng, 2007), is developed to formalize a design problem described in natural language.

Axiomatic theory of design modeling is a logical tool for representing and reasoning about object structures (Zeng, 2002). It provides a formal approach that allows for the development of design theories following logical steps based on mathematical concepts and axioms. The primitive concepts of universe, object, and relation are used in the axiomatic theory of design modeling, based on which two axioms are defined in the axiomatic theory of design modeling.

[Axiom1] Everything in the universe is an object.

[Axiom 2] There are relations between objects.

Structure operation is developed in the axiomatic theory of design modeling to model the structure of complex objects. Structure operation, denoted by the symbol \oplus , is defined by the union (\cup) of an object and the interaction (\otimes) of the object with itself.

$$\oplus O = O \cup (O \otimes O) \quad (3.3)$$

where $\oplus O$ is the structure of an object O . Both the union and interaction are specific relations between objects.

Since the object O may include other objects, Eq. (3.3) indeed implies a recursive representation of an object.

It is shown that both design requirements and product descriptions, as illustrated in Figure 6, are implied in product system (Zeng, 2004a). A product system is defined as the structure of an object (Ω) including both a product (S) and its environment (E).

$$\oplus \Omega = \oplus (E \cup S) = (\oplus E) \cup (\oplus S) \cup (E \otimes S) \cup (S \otimes E), \forall E, S [E \cap S = \Phi], \quad (3.4)$$

where Φ is the object that is included in any object. $\oplus E$ and $\oplus S$ are structures of the environment and product, respectively; $E \otimes S$ and $S \otimes E$ are the interactions between environment and product. A product system can be illustrated in Figure 7.

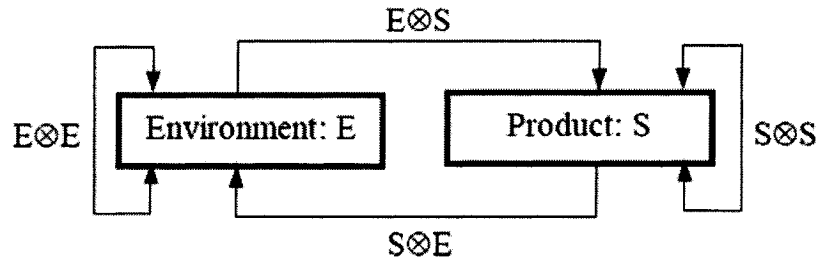


Figure 7 Product system

Since environment as well as product may have components, structures $\oplus E$ and $\oplus S$ can be further decomposed into the structures of these components as well as their mutual

interactions according to the definition of structure operation. Eq. (3.4) indeed presents a recursive structure of a product system. Therefore, the structure operation provides a mechanism that can flexibly represent the structure of any complex object.

It is straightforward to see that the product description is implied in the product system $\oplus \Omega$ while the product requirements are not explicitly given in Eq. (3.4). It is shown that all the product requirements in a design problem are imposed by the product environment (E) in which the product is expected to work (Zeng, 2004b). Product environment is the driving force of a design process. Product requirements are part of interactions between product and environment. Design constraints belong to the relations from environment E to the product S ($E \otimes S$) whereas product functions belong to the relations from product S to environment E ($S \otimes E$).



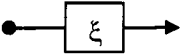

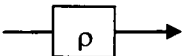
3.4 Recursive object model (ROM)

To formalize design problems, a formalized structure is needed to represent the problems, based on which it is possible to convert a design problem to an engineering problem that can be solved by a mathematical method.

Recursive object model (ROM) language is proposed to illustrate the modeling components graphically in order to formalize design problems using the axiomatic theory of design modeling. ROM is a graphic representation of linguistic structure, derived from axiomatic theory of design modeling. Zeng designed a formalized structure and specified graphic symbols to represent objects and relations (Zeng, 2007). These objects and

relations can be mapped in the design problem described by natural language. Table 1 shows the graphic symbols in the ROM.

Table 1 Symbols in ROM

Type		Graphic Representation	Description
Object	Object		Everything in the universe is an object.
	Compound Object		It is an object that includes at least two objects in it.
Relations	Constraint Relation		It is a descriptive, limiting, or particularizing relation of one object to another.
	Connection Relation		It is to connect two objects that do not constrain each other.
	Predicate Relation		It describes an act of an object on another or that describes the states of an object.

Chapter 4

Asking the Right Questions to Elicit Product Requirements

In this chapter, a generic inquiry process is proposed to elicit product requirements. The goal of this generic inquiry process is to elicit precise and complete requirements for a design task. To illustrate how this process works, the design of a cognitive experiment by using EEG system is given step by step.

4.1 General steps for product requirements elicitation

Any design process starts with a description of a design task, which is often described in natural language by customers. Therefore, the design task is customer oriented and may be ill-defined while the requirements elicitation process should generate formal and structured descriptions, which are engineering oriented. Asking the questions is an effective way to identify the customer's real intent and to define a relatively more complete list of product requirements.

We have proposed a generic inquiry process for eliciting product requirements (Wang and Zeng, 2007a; Wang and Zeng, 2007b), which is shown in Figure 8. The process can be divided into the eight steps as follows. An algorithmic description is given later in this section followed by a detailed case study.

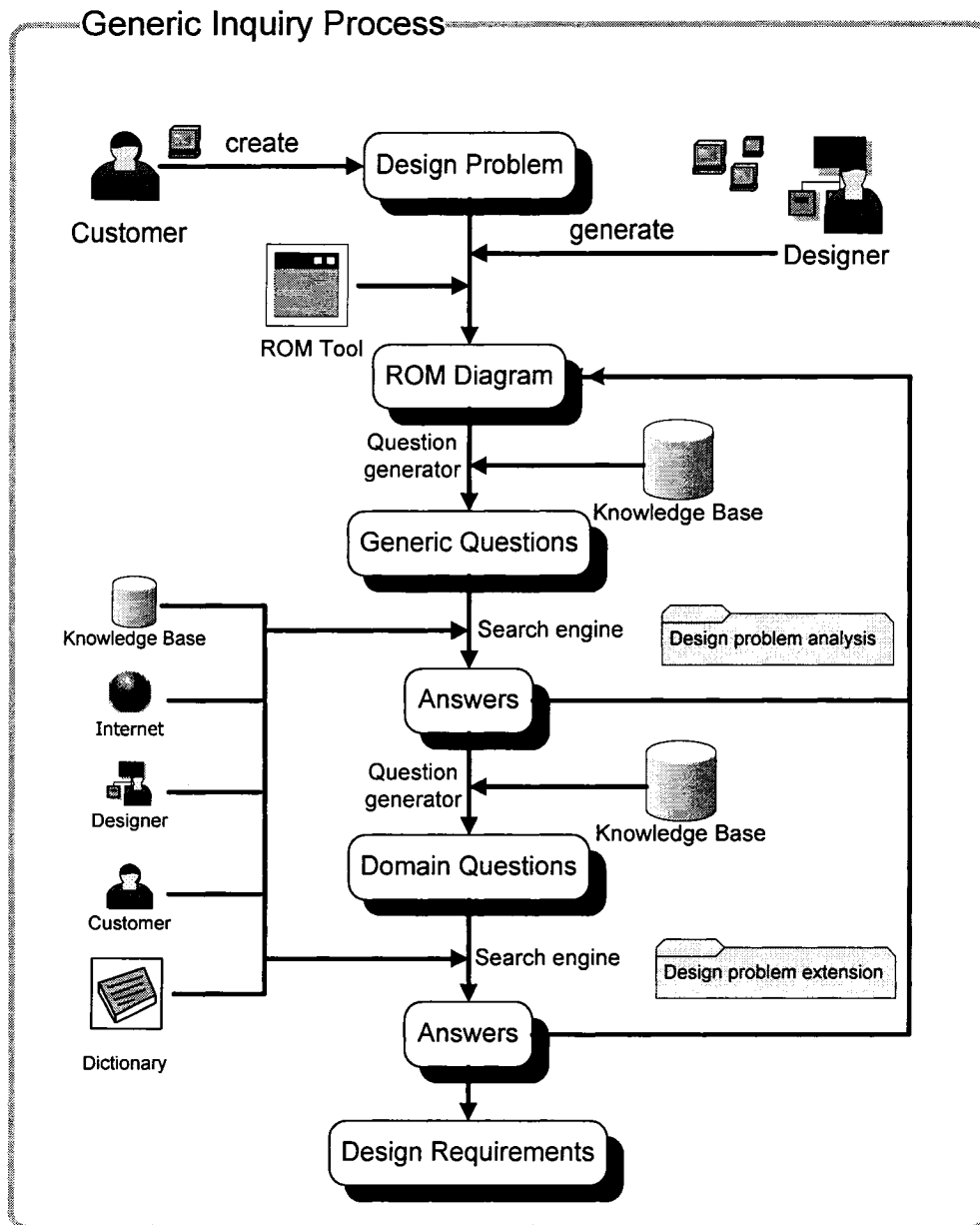


Figure 8 Generic inquiry process for requirements elicitation

Step 1: Create ROM diagram.

Designer transforms the original design problem described by natural language into a ROM diagram using the ROM analysis tool. This will enable the designer to understand the design problem more clearly.

Step 2: Generate generic questions.

In this step, designer analyzes each object in the ROM diagram to find out the objects that need to be identified or clarified further and generates some candidate questions based on a set of predefined rules and question template. Then these questions will be chosen to ask. These questions will help customers to understand and to clarify their real intent.

Step 3: Collect answers.

Designer collects the answers to the questions which are generated from Step 2 by looking up dictionary or knowledge base, searching on internet, or collecting from the customer.

Step 4: Repeat steps 1 to Step 4 until no more generic questions can be asked.

The answers collected in Step 3 are analyzed iteratively as new ROM diagrams at Step 1. Then these new ROM diagrams are identified or clarified by asking questions at Step 2. Thereafter, ROM diagrams are merged into the original ROM diagram in the ROM tool based on a set of predefined rules.

At the end of the step, the explicit design problem description is extended further. The recursive process will be shown more clearly in the algorithmic description of the procedures.

Step 5: Generate domain specific questions.

The explicit design problem description above may not exactly and completely define the design problem. Designer needs to elicit more implicit environment information about the design problem. In this step, designer analyzes the relationships between the objects in the updated ROM diagram and generates a set of questions that should be answered at each stage of the design. Meanwhile the sequence for asking these questions should be determined automatically or manually based on a set of predefined rules.

Step 6: Collect answers to the questions generated in Step 5.

This step is similar with Step 3.

Step 7: Repeat Step 1 to 7 until no more domain questions can be asked.

To collect complete domain requirements and analyze the implicit domain requirements iteratively in Step 1 to 7 may ensure that the domain requirements are accurate.

Step 8: Output the updated design problem description.

The process presented above can be alternatively represented by the following algorithmic process described in pseudocode. The whole process is made up of two sub-processes, which are design problem analysis and design problem extension.

//Function of generic inquiry process for requirements elicitation

//Generic Process - Input: text; Output: ROM

ROM GenericProcess(text)

{

//Analyze design problem through generic questions

ROM \leftarrow DesignProblemAnalysis (text);

// Analyzed design problem through domain specific questions

ROM \leftarrow DesignProblemExtension (ROM);

}

The design problem analysis is a recursive process by asking the generic questions to get the real intent from customer. This process will analyze text described in natural language to determine questions to be asked and collect answers, which will be analyzed iteratively within the process. Finally a merged ROM diagram can be generated with a more detailed requirements description. The following pseudocode describes the process.

//Sub-function of design problem analysis by asking generic questions

// Input: text; Output: ROM

ROM DesignProblemAnalysis (text)

{

//Generate the ROM diagram for the design problem text

ROM \leftarrow ROMAnalysis (text);

// Object analysis to determine objects to be asked

```

ROM ← ObjectAnalysis (ROM);

// Generate question list

Q_list ← GenericQuestionGeneration (ROM);

//Ask the questions

if (Q_list is not empty)
forall (Q in Q_list)
{
    //Collect answer to question Q

    answer ← GetAnswer(Q);

    //Call ROM analysis recursively

    ROM1 ← DesignProblemAnalysis (answer);

    //Merge ROM1 to ROM

    ROM←ROMMerge (ROM, ROM1);

}

else return ROM;

}

```

The other sub-process, design problem extension, is also a recursive process, in which the domain specific questions are asked to get the complete requirements. Those questions are generated based on an all-around classification of requirements embedded in the whole product life cycle. In the same way, the answers will be analyzed iteratively within the process of design problem analysis. The following pseudocode describes the process.

//Sub-function of design problem extension by asking domain questions

// Input: ROM; Output: ROM

ROM DesignProblemExtension (ROM)

```
{  
    // Call DomainQuestionGeneration to get question list  
    Q_list ← DomainQuestionGeneration (ROM);  
    //Ask the questions  
    if (Q_list is not empty)  
    {  
        forall (Q in Q_list)  
        {  
            //Collect answer to question Q  
            answer ← GetAnswer(Q);  
            //Call ROM analysis recursively  
            ROM2 ← DesignProblemAnalysis (answer);  
            //Merge ROM2 to ROM  
            ROM←ROMMerge (ROM, ROM2);  
        }  
    }  
    else return ROM;  
}
```

The details of ROMAnalysis and ROMmerge are given in a disclosure to the Office of Research at Concordia University for a patent application (Zeng et al., 2007). My focus

in this thesis is on the processes for question generation. The following subsections will discuss the main steps of those two processes in details.

4.2 Asking the generic questions

The sub-process of asking generic questions is shown in Figure 9. This inquiry process is designed to get the real intent of design.

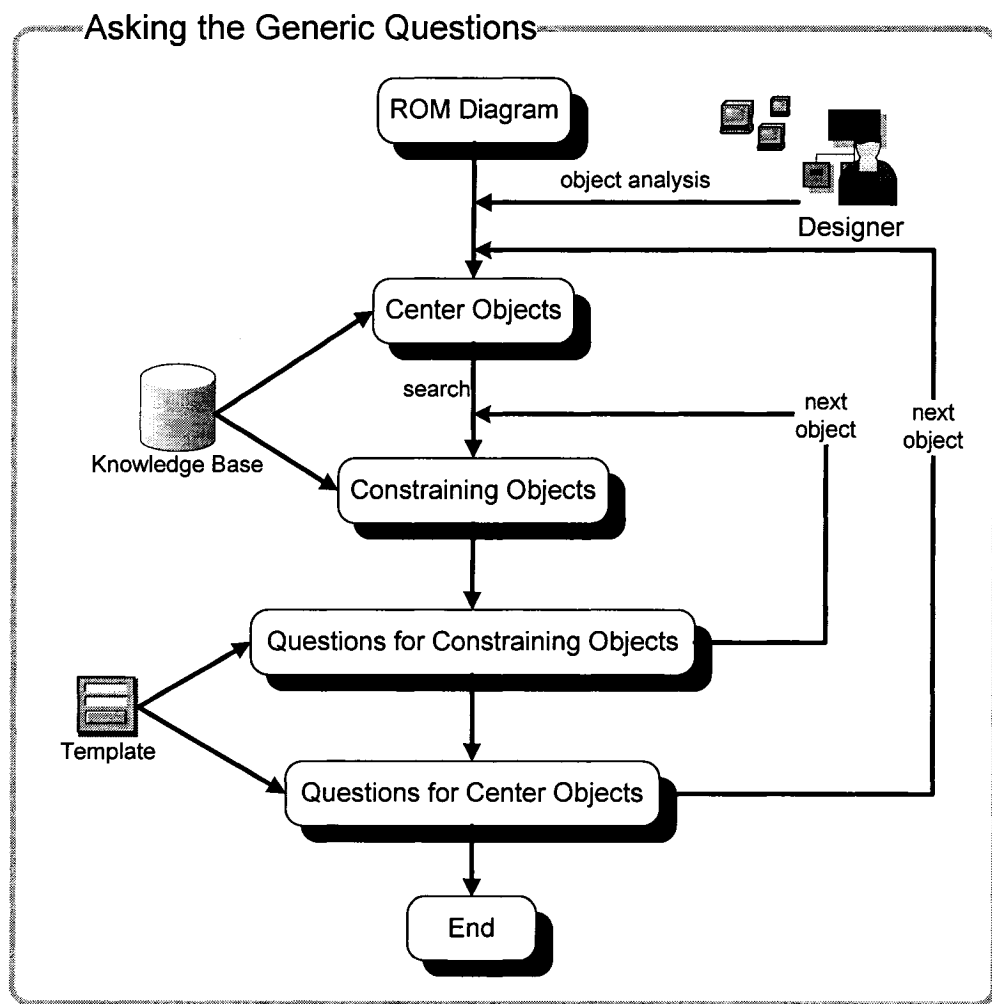


Figure 9 Asking the generic questions

In this step, each object in a ROM diagram is analyzed and categorized as a center object or a constraining object that needs to be identified or clarified further. The whole ROM diagram is a directed graph with objects and relations from one object to another object. The graph can be represented as $G = \langle V, E \rangle$, where V is a set of objects and E is a set of edges between the objects, which are indeed relations. This representation enables us to apply algorithms from graph theory to our concerned problem. The rules Rule1 and Rule2 listed in Table 2 can be applied to determine which objects should be extended first.

Table 2 Rules for objects analysis

• Rule1	Before an object can be further defined, the objects constraining them should be refined.
• Rule2	An object with the most undefined constraints should be considered first.

The pseudocode for the algorithm of asking generic questions is shown in the following.

```

//Algorithm for asking generic questions
// Input: ROM; Output: QuestionList
QuestionList AskGenericQuestion (ROM)
{
    // Get the ObjectList of ROM diagram
    ObjectList ← GetObject (ROM)
    if (Q_list is not empty)

```

```

{
    forall ( v in ObjectList)
    {
        // Get the number of relations Nr to V
        Nv ← GetNumRelation (ObjectList);
    }
    // Sort ObjectList in terms of Nv
    ObjectList ← ObjectSort (ObjectList, Nv);
    // Generate questions according to sorted ObjectList
    QuestionList ← GenerateQuestion (ObjectList);
}
else return ErrorMessage;
}

```

Here an example is given to show how the algorithm and rule R1 and rule R2 work. The design problem is to design an experiment to quantify people's mental stress level by using an EEG system. In the ROM diagram of the text in Figure 14, based on R1 and R2, the most undefined constraining objects are *EEG system* and *people's mental stress* should be clarified first.

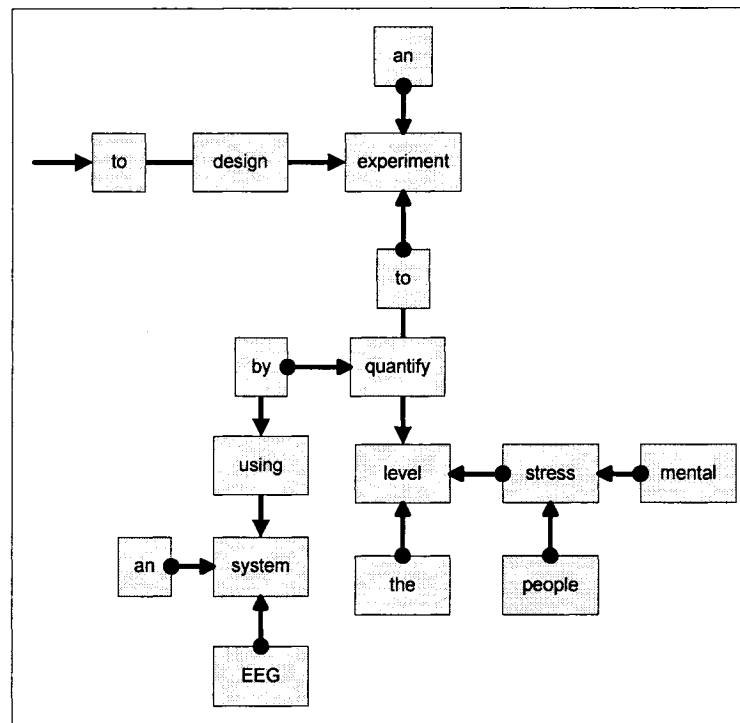


Figure 10 ROM diagram for EEG experiment design

Then the system generates some candidate questions based on a set of predefined template from the knowledge base and designer can select some of these questions to ask based on a question template. Table 3 shows some examples of question templates.

In the ROM linguistic analysis, objects are distinguished as part of speech. As the primary objects, three types of nouns: concrete noun, proper noun, and abstract noun are shown in Table 4.

Table 3 Questions template for object analysis

T1	For a concrete, proper, or abstract noun N	Question: What is N ?
T2	For a noun naming a quantity Q of an object N , such as height, width, length, capacity, and level, such as height, width, length, capacity, and level	Question: How many / much / long / big / ... is the Q of N ?
T3	For a verb V	Question: How to V ? Or Why V ?
T4	For a modifier M of a verb V	Question: Why V M ?
T5	For an adjective or an adverb A	Question: What do you mean by A ?
T6	For a relation R that misses related objects	Question: What (who) R (the given object)? Or (the given object) R what (whom)?

Table 4 Classification of nouns

• Concrete noun	A thing that one can perceive through ones physical senses such as touch, sight, taste, hearing, or smell.
• Proper noun	That represents the name of a specific person, place, or thing. It is always written with a capital letter, such as ROM, ISO.
• Abstract noun	That refers to states, events, concepts, feelings, qualities, etc., that have no physical existence, and is the opposite of a concrete noun, for example, “freedom”, “happiness”, and “idea”.

Therefore, in the EEG example, two questions should be asked as following:

- What is the EEG system in your design problem?
- What is the people's mental stress?

4.3 Asking the domain questions

To collect complete list of product requirements, in the generic inquiry process we propose a sub-process of asking the domain questions. The questions are generated in terms of product life cycle and classification of requirements.

4.3.1.1 The product life cycle (PLM)

Product life cycle is usually studied in terms of the phases that occur according to the time sequence. Different projects may have different life cycles. For example, for a mechanical product, the life cycle is divided into seven kinds of events, which are design, manufacture, sales, transportation, use, maintenance, and recycle, as shown in Figure 11 (Chen and Zeng, 2006), whereas software has its own phases. A typical life cycle of software includes requirements phase, specification phase, design phase, implementation phase, integration phase, maintenance phase and retirement (Schach, 2002). These steps can be shown in the waterfall model of Figure 12. In addition, for an experimental psychology design, the life cycle includes experiment preparation, experiment design, data collection, data analysis, and experiment verification (Kantowitz *et al.*, 2001).

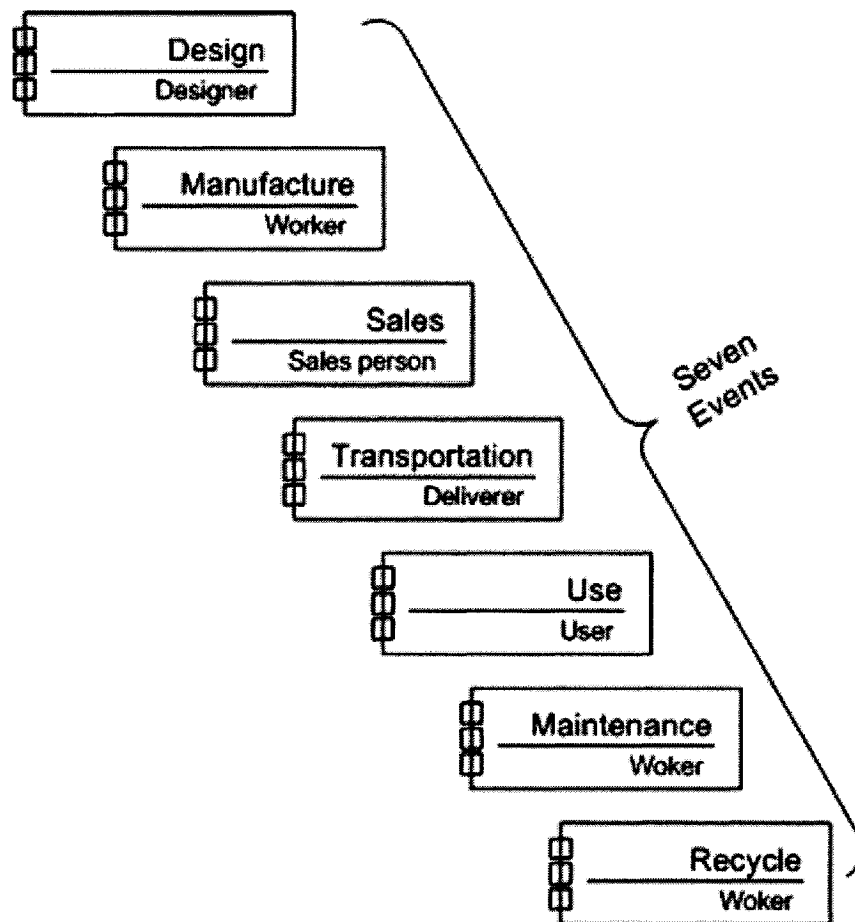


Figure 11 Seven events in product life cycle (Chen and Zeng, 2006)

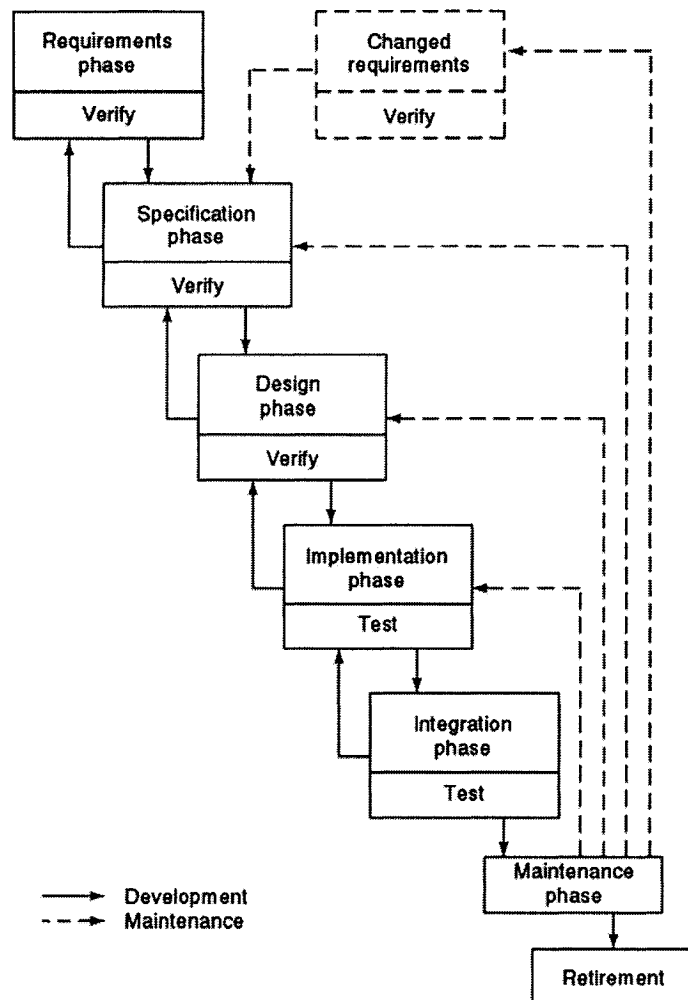


Figure 12 Waterfall model in software engineering (Schach, 2002)

Despite the differences of product types, those events are building blocks of entire product life cycle. Chen and Zeng summarized rules for the events in product life cycle. They presented that at any time point of the mechanical product life, one or some of those seven events may occur simultaneously or alternatively (Chen and Zeng, 2006). All the requirements are classified into those seven kinds of events so that various requirement

providers are able to concentrate on their respective parts, which are associated with their relevant environment.

4.3.1.2 Ranking of priority of requirements

On the other hand, in the whole life cycle, the source of product requirements varies and the number of requirements for a single product may be huge. It is usually challenging to design a product to satisfy all the requirements. Hence, it is necessary to rank all the requirements so that designers can easily know which requirements have higher priority.

Chen and Zeng (Chen and Zeng, 2006) gave a useful classification of requirements, which provides a complete requirements list. In Figure 13, the product requirements are categorized into eight levels: natural laws, social law and regulations, technical limitation, cost, time and human resource, basic functions, extended functions, exception control level, and human-machine interface. In this pyramid-like model, those requirements at the lower levels have higher priority in developing a design solution. And those meeting the requirements at the highest level are called high usability products.

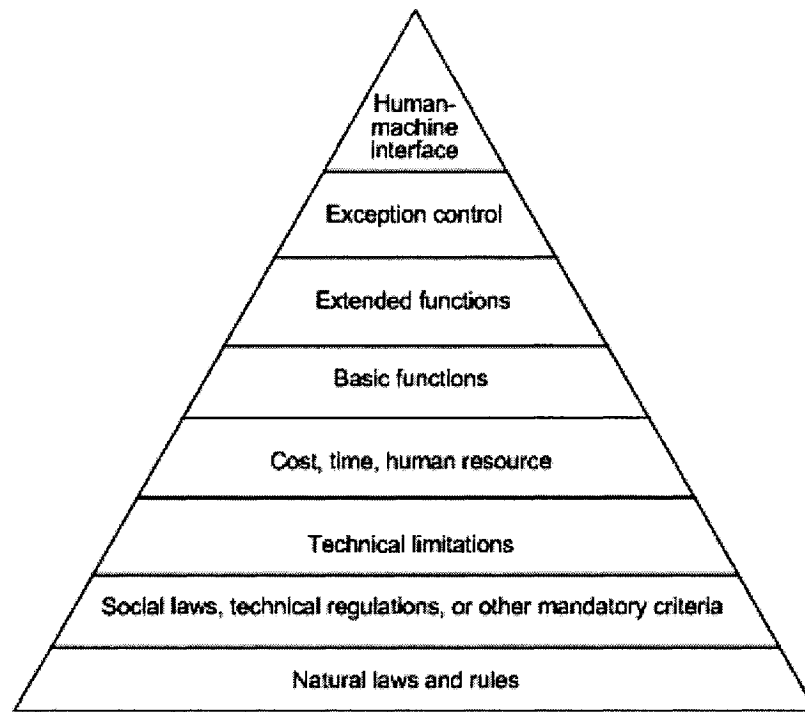


Figure 13 Eight levels of requirements (Chen and Zeng, 2006)

In this model, higher-level requirements can be considered after lower-level requirements are satisfied. Basically, this pyramid-like model can be divided into two major groups: non-functional requirements, and functional requirements. The lower four: natural law and rules, social law, regulations, technical limitations, cost, time, and human resource level are usually non-functional requirements. The upper four: basic functions, extended functions, exception control, human-machine interface are usually functional requirements.

These eight levels of product requirements are also related to product environment. In this case, the product environment can be partitioned into natural, built, and human environments (Zeng, 2004a). The highest four levels of product requirements come from human environment. They serve for the purposes of human use of the product. The

lowest level of product requirements comes from natural environment. The rest is the result of built environment.

4.3.1.3 Generation of domain specific questions

Based on the product life cycle and the requirements classification (Chen and Zeng, 2006), the sub-process of asking the domain questions is shown in Figure 14. A question for product life cycle is first asked to identify the related stages in which the product is involved. Then for each stage, questions are generated in terms of environment components on their requirement level. A template is proposed to help generate the questions and determine the sequence of questions. Table 5 shows the rules for asking the domain specific questions.

Table 5 Questions generation rules for Step 5

• Rule3	First to ask: What is the life cycle of the product to be designed?
• Rule4	Ask questions about the natural, built, and human environment about each stage of the lifecycle of the product.
• Rule5	The sequence for asking questions is determined by the levels of requirements in Figure 3 so that those requirements at the lower levels have higher priority and can be asked earlier.
• Rule6	Ask questions about the answers from Rule 1 and Rule 2 by applying the rules related to Step 2.

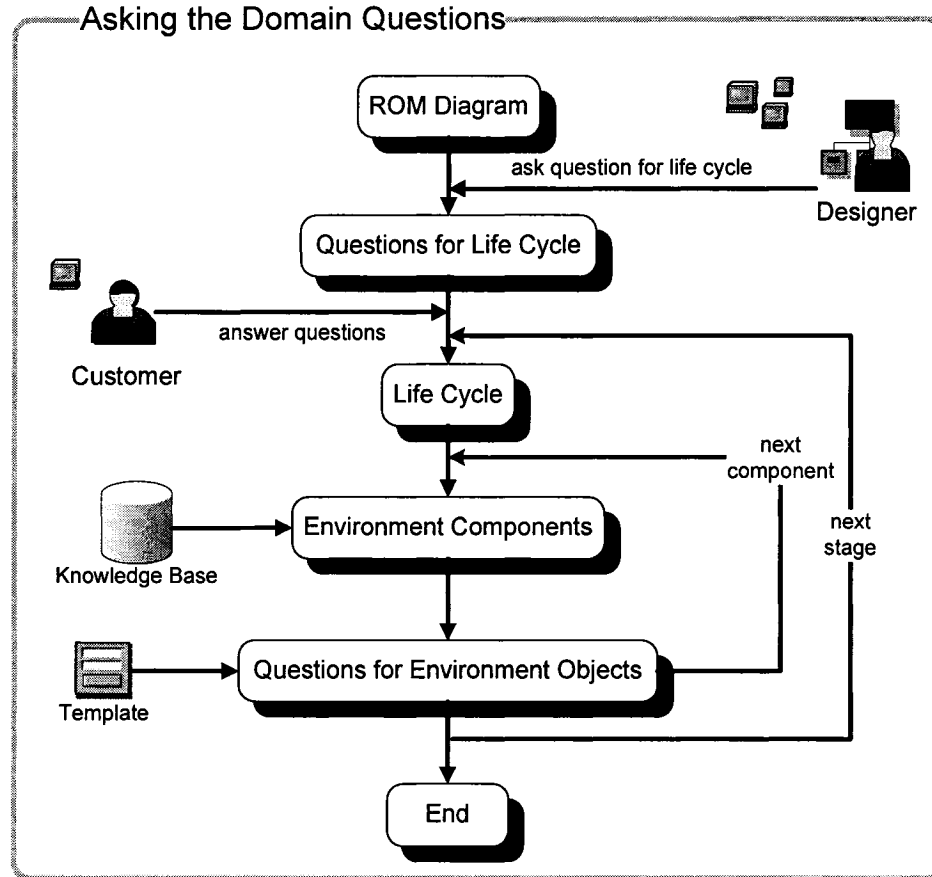


Figure 14 Ask domain specific questions

4.4 Case study of EEG experiment design

An example of EEG experiment design is used to support the proposed generic process. In this case study, the task of this problem is to design an experiment to quantify people's mental stress level by using an EEG system. We follow the generic formalization process we provided as below.

Step 1: Create ROM diagram based on design problem description. The ROM diagram is shown in Figure 15.

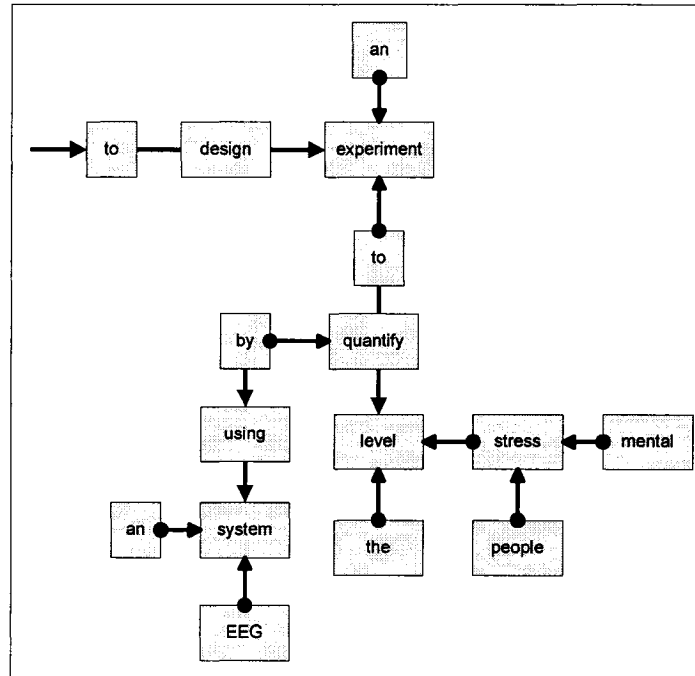


Figure 15 ROM diagram for Step 1

Step 2: Generate Questions based on ROM diagram.

In this step, designer generates some candidate questions based on a set of predefined rules on Table 2 and selects some of these questions to ask based on a question template such as those given in Table 3.

Based on Rule1 and Rule2, the EEG system and people's mental stress should be clarified first. Therefore, according to question template for object analysis, questions Q1 and Q2 according to T1 are asked first in this step. And then Q3 referred to T2, Q4 referred to T3 are asked.

To get customer's real intent or a more innovative solution, another question can be asked as Q5. The questions list for Step2 is listed in Table 6.

Table 6 Questions for Step 2

• Q1	What is the EEG system in your design problem?
• Q2	What is the people's mental stress?
• Q3	How many levels are there for people's mental stress?
• Q4	How to use an EEG system?
• Q5	Why do you want to use EEG system in your experiment?

Step 3: Collect answers for the generated questions.

Designer collects the answers to the questions of step 2 by looking up dictionary or knowledge base, searching on internet, or collecting from customer. The answers are listed in the Table 7.

Step 4: Repeat steps 1 to 4 to analyze the answers collected in Step 3 until no more questions can be asked.

Each answer should be gone through from Step 1 to Step 4. The following details show the ROM analysis one by one answer.

Table 7 Answers for Step 3

• A1	EEG system is a system to record human brain waves during human activities.
• A2	People's mental stress is the sum of physical and mental responses to an unacceptable disparity between real or imagined personal experience and personal expectations.
• A3	There is no specific classification about people's mental stress in the current research. That needs to be researched further.
• A4	To find the mathematical relationship between the patterns of brain waves and human performances.
• A5	The EEG can detect changes in electrical activity in the brain on a millisecond-level.

A1: EEG system is a system to record human brain waves during human activities.

The ROM diagram for the A1 is shown in Figure 16.

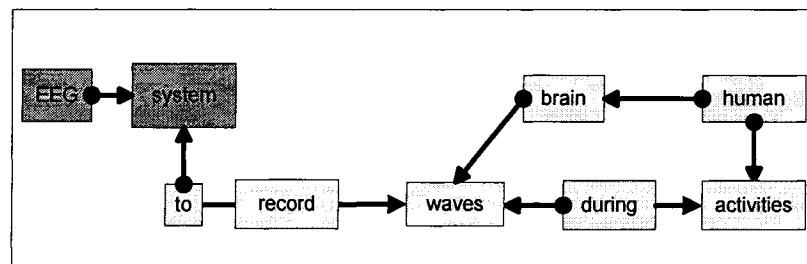


Figure 16 ROM diagram for A1

A2: People's mental stress is the sum of physical and mental responses to an unacceptable disparity between real or imagined personal experience and personal expectations.

The ROM diagram for the A2 is shown in Figure 17.

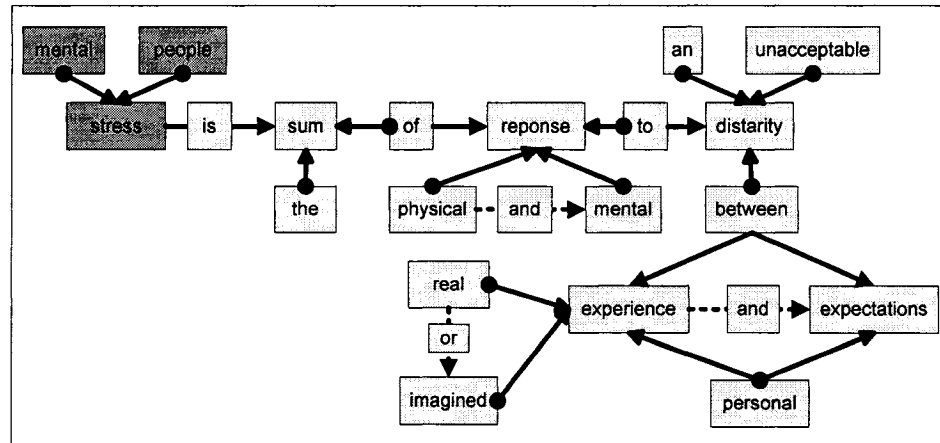


Figure 17 ROM diagram for A2

A3: There is no specific classification about people's mental stress in the current research. That needs to be researched further.

A4: To find the mathematical relationship between the patterns of brain waves and human performances.

The ROM diagram for the A4 is shown in Figure 18.

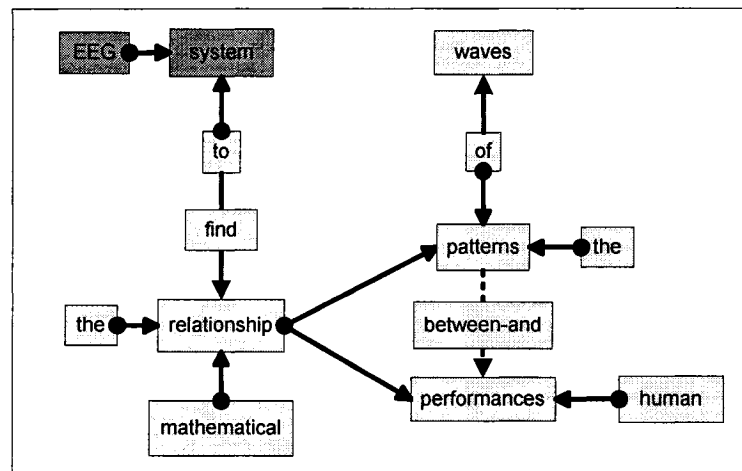


Figure 18 ROM diagram for A4

A5: The EEG can detect changes in electrical activity in the brain on a millisecond-level.

The ROM diagram for the A4 is shown in Figure 19.

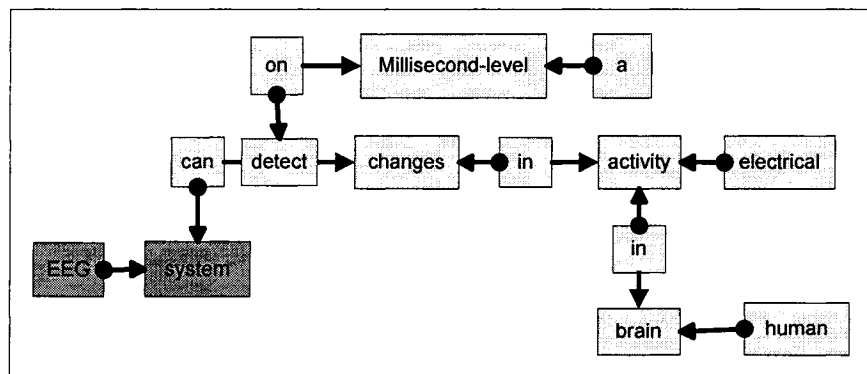


Figure 19 ROM diagram for A5

In this case study, up to Step 4 there is no more questions to ask. The explicit design problem description is extended further at the end of the step. The new ROM diagram is shown in

Figure 20 Merged ROM diagram for Step 4

To identify the implicit environment and complete requirements of the design problem based on a set of predefined rules in Table 5 some questions will be selected to ask. These questions shown in Table 8 should be answered at each stage of the design.

Table 8 Questions for Step 7

• Q6	What is the life cycle of the experiment to be designed?
• Q7	What standards should the experiment conform to?
• Q8	How long will the experiment last?
• Q9	Who are the subjects in the experiment?
• Q10	Who are the experimenters in the experiment?

Step 6: Collect answers for the questions generated in Step 5.

The collected corresponding answers are listed as Table 9:

Table 9 Questions for Step 8

• A6	The life cycle of the experiment includes generating problem, design, collecting data, and analyzing data.
• A7	The experiment should follow the related health safety standards.
• A8	The experiment can last about 1 hour.
• A9	The subjects of the experiment are persons with coordinative education.
• A10	The experimenters of the experiment are persons with related knowledge.

Step 7: Repeat Step 1 to 7 until no more domain questions can be asked.

Each answer should be analyzed in ROM diagram from Step1 to Step 7 and generated ROM diagrams are merged into the previous one. A part of the final ROM diagram is shown in Figure 21.

Step 8: Output the updated design problem description.

Based on the last ROM diagram generated in the whole generic formalization process, we get a part of final requirements by translating the ROM diagram into natural language. The descriptions are listed as below.

The task is to design an experiment to quantify the level of people's mental stress by using an EEG system. In the experiment, people's mental stress is tested on physical and mental responses to an unacceptable disparity between real or imagined personal experience and personal expectations. The EEG system can record human brain waves during human activities and detect changes in electrical activity in the brain on a millisecond-level, and finally to find the mathematical relationship between the patterns of brain waves and human performances.

The life cycle of the experiment includes generating problem, experiment design, collecting data, and analyzing data. The experiment can last about 1 hour and should follow the related health safety standards. The subjects of the experiment are persons with relevant education. The experimenters of the experiment are persons with related knowledge.

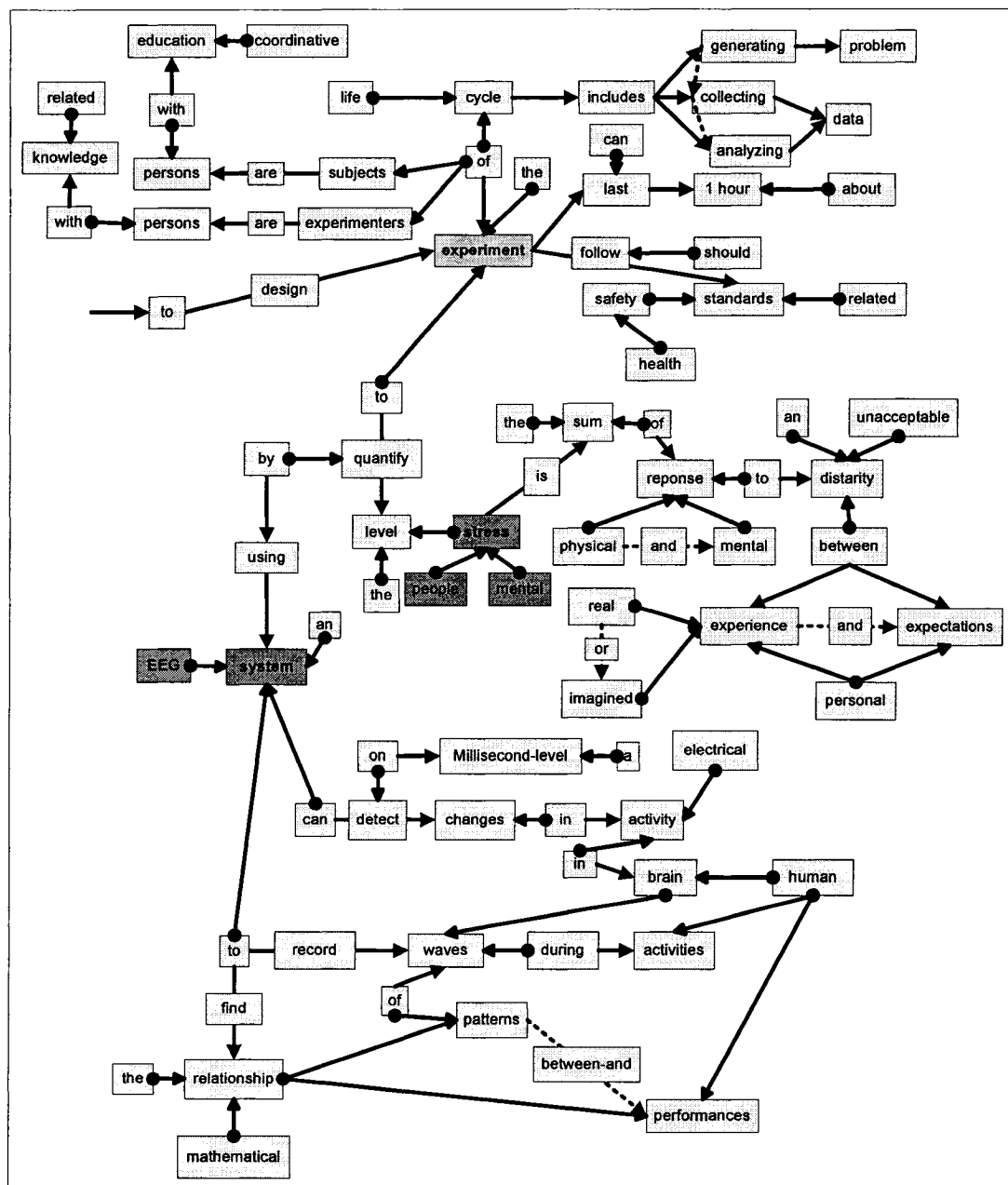


Figure 21 Part of the final ROM diagram

Chapter 5

Software Prototype for Eliciting Product Requirements

The quality of design can be improved by a better understanding of the requirements definition and elicitation process. In Chapter 3, the method of Environment Based Design, the classification of requirements and Recursive Object Model are introduced. In Chapter 4, a generic questioning process is proposed to elicit the product requirements. Since the processing of ROM diagrams could be very time consuming, the process needs a design tool to facilitate designer to analyze design problem and generate questions. Meanwhile, a convenient communication platform is also essential for the smooth transitions of questions and answers between designer and customer. Computer aided platform and tool can improve the precision and efficiency of the process.

5.1 Problem formulation

In this thesis research, a software prototype was developed to help designers build a generic formalized product system represented by the ROM language from a design problem described in natural language. Meanwhile, the software prototype will be used to validate the generic process that we have proposed in this thesis. The input of this software system is a design problem described in English by customers whereas the corresponding ROM diagram is the output. Customers use the software to present the design problem and answer the questions asked by designer to help elicit the precise requirements. Designers use the software to analyze the design problem and generate the final requirements.

In this chapter, I will introduce the system architecture, primary interfaces and the implementations of using the software prototype. A case study of rivet setting tool design example is selected to show how the software prototype works.

5.2 System architecture

We have implemented a software system named Web-based Collaborated Requirements Management System. The software system contains five modules: system management, design problem generation, design problem analysis, design problem extension and document generation. The architecture of this system is shown in the Figure 22.

There are three actors in the system: the administrator executes system management; the customer generates design problem and helps analyze design problem; and the designer implements the primary functions including design problem analysis and requirements elicitation. All the users must be given authentication and authorization to use the system. And all the work results should be saved up in related documents. The outer recursive process is represented in the transfer between design problem analysis and design problem extension, in which the latter module deals with the ROM diagram from the former module and the extended answers generated in the latter module must be analyzed in the former module. At last the engineering specifications are generated in the process for concept generation.

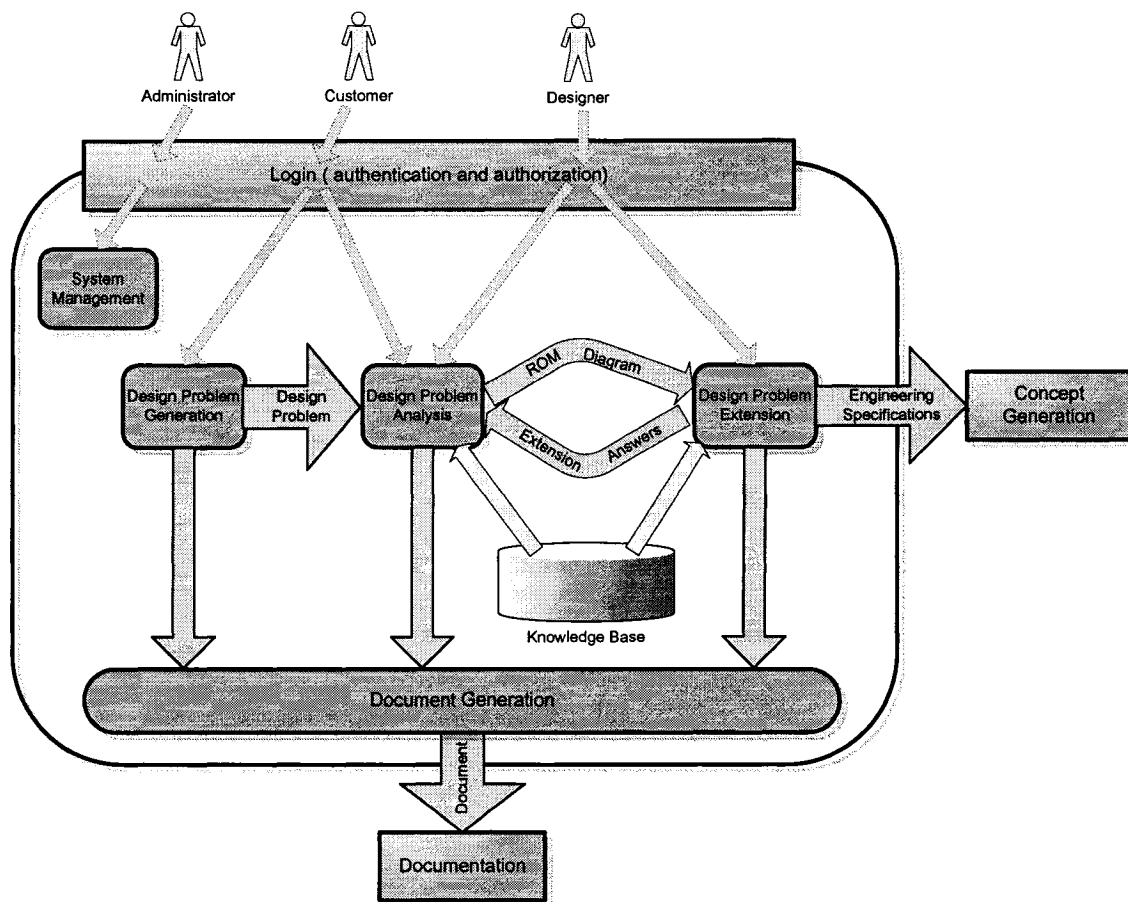


Figure 22 System architecture of the software prototype

According to the generic requirements elicitation process, the module of design problem analysis corresponds to the recursive process from Step 1 to Step 4 as shown in Figure 23. The module includes four core sub-modules: ROM generation, object analysis, question generation, and answer collection. The inner recursive process is shown in this module. The collected answers must be analyzed within the module.

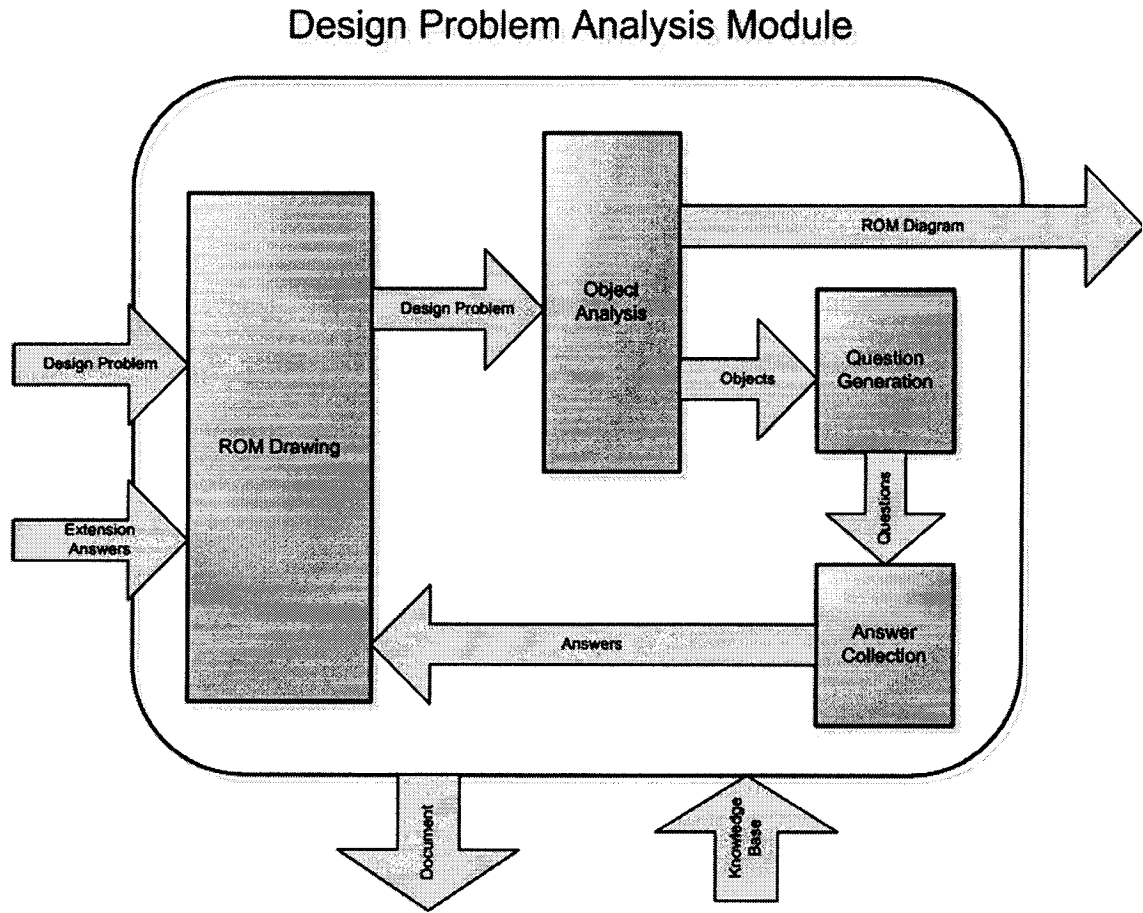


Figure 23 Module of design problem analysis

Meanwhile, the module of design problem extension corresponds to the recursive process from Step 5 to Step 7 as shown in Figure 24. The module includes environment analysis, question generation, and answer collection. The question generation in module of design problem extension is based on different templates compared with that in module of design problem analysis. The two answer collection modules are similar in the two main modules.

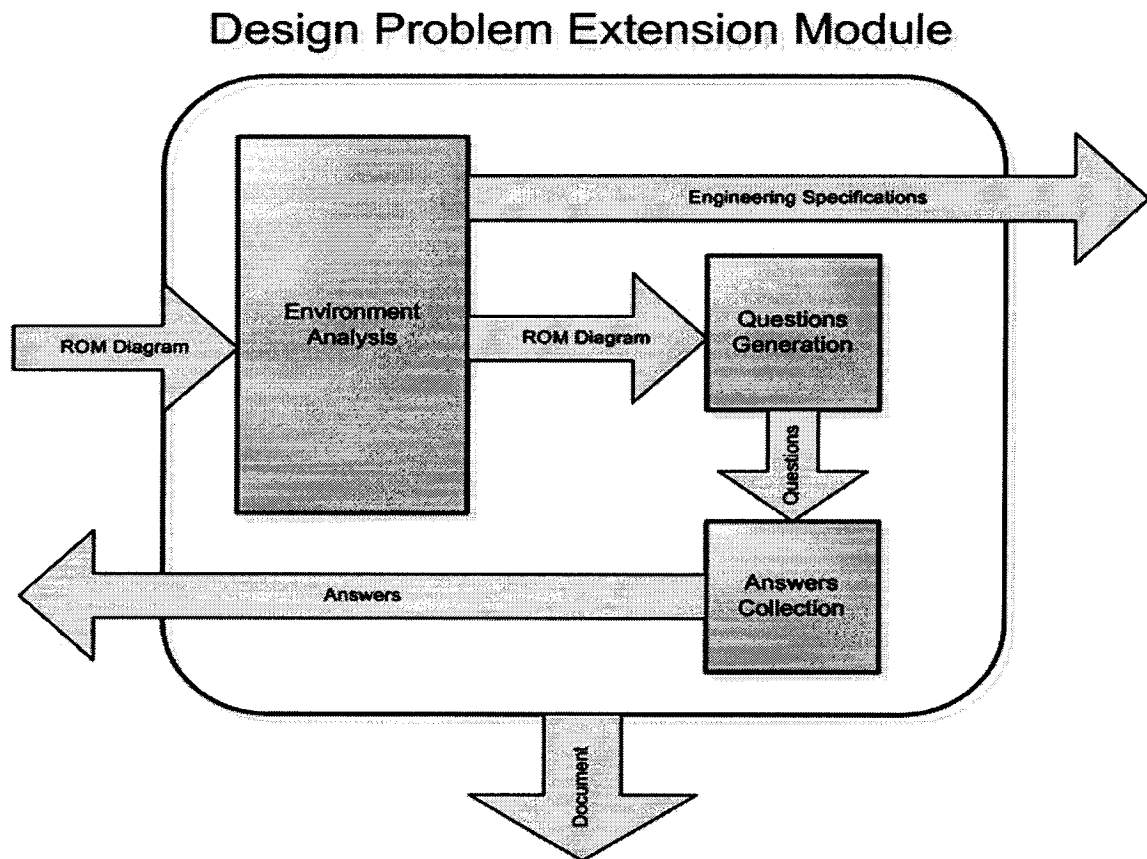


Figure 24 Module of design problem extension

5.3 System interfaces

In this section, some main interfaces in screen shots are given to show the operation process of the software prototype. The main software is web-based framework. Except two part of them, the details of ROM analysis shown in Figure 28 and question generation shown in Figure 29 as requirements elicitation tools in windows application given in a technical report, based on which we are filing a disclosure to the Office of Research at Concordia University for a patent application (Zeng et al., 2007).

The user login and authentication interface is shown in Figure 25. Administrator, design and customer can log in it and enter the system.



Figure 25 Interface for login and authentication

Figure 26 shows the interface of homepage after login the system. The difference for customer and designer is that customer has the access to create a design problem and enter the created project to assist for problem analysis by answering the questions asked by designer. The designer can search projects and access the selected projects for problem analysis.

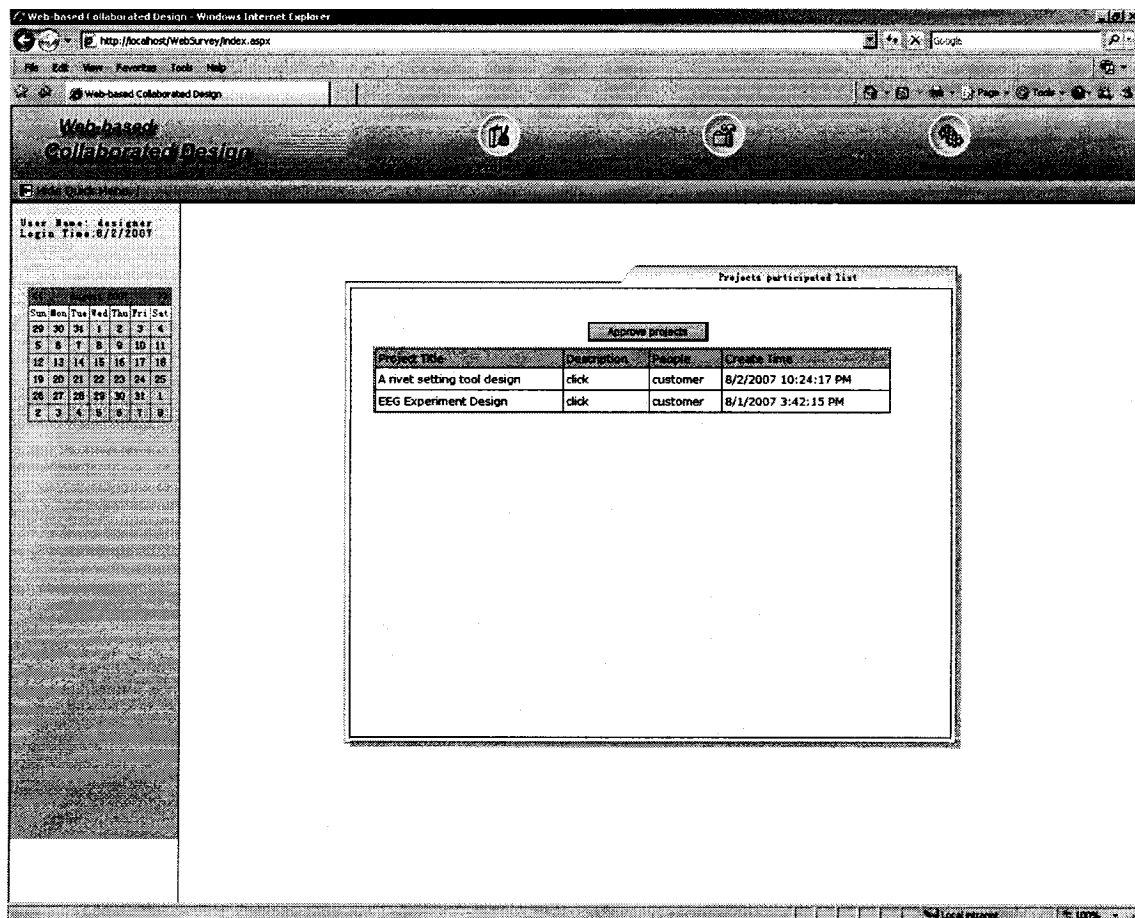


Figure 26 Interface for homepage

The interface for design problem input is shown in Figure 27. Customer can create a design problem here and save it.

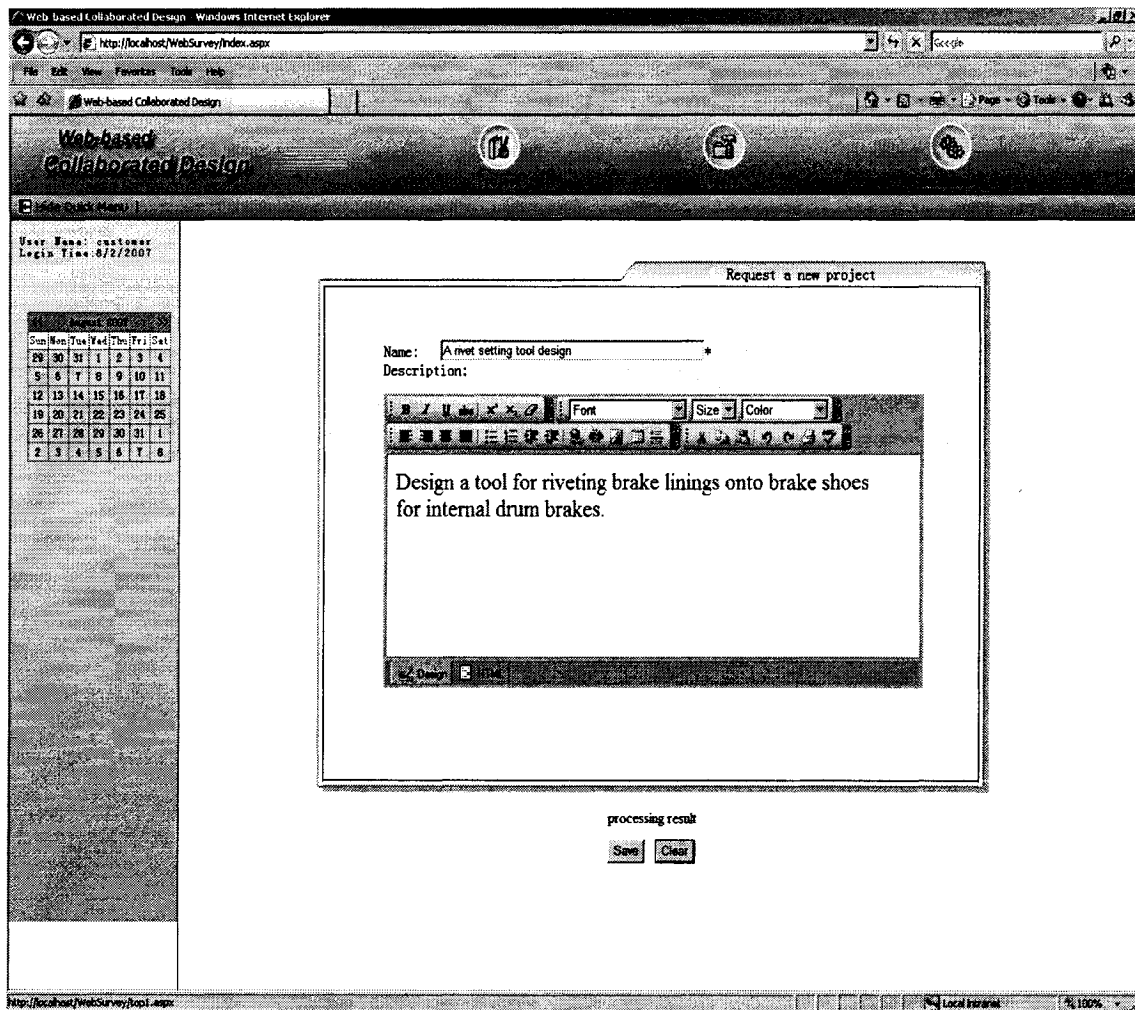


Figure 27 Interface for design problem input

Figure 28 is the screen shot of ROM analysis (Zeng et al., 2007). Designer can transfer the text of design problem descriptions or collected answers into ROM diagrams. Many functions like ROM create, ROM edit, ROM merge, ROM save are embedded in the tool.

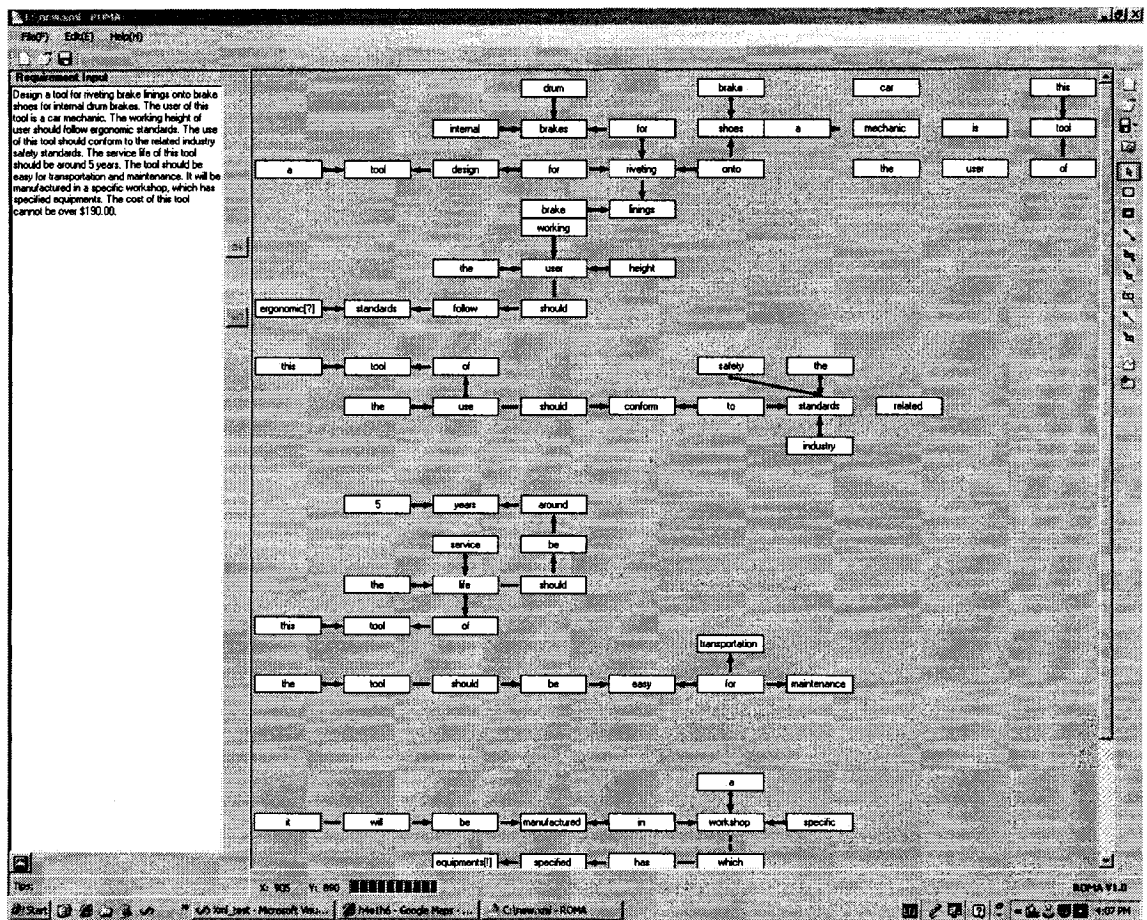


Figure 28 Interface for ROM drawing

The question generation tool is shown in Figure 29 (Zeng et al., 2007). Designer uses the tool to analyze the ROM diagram and selects some questions from those candidate questions created by the system based on predefined rules and template. Also designer can input questions manually and change the questions with flexibility.

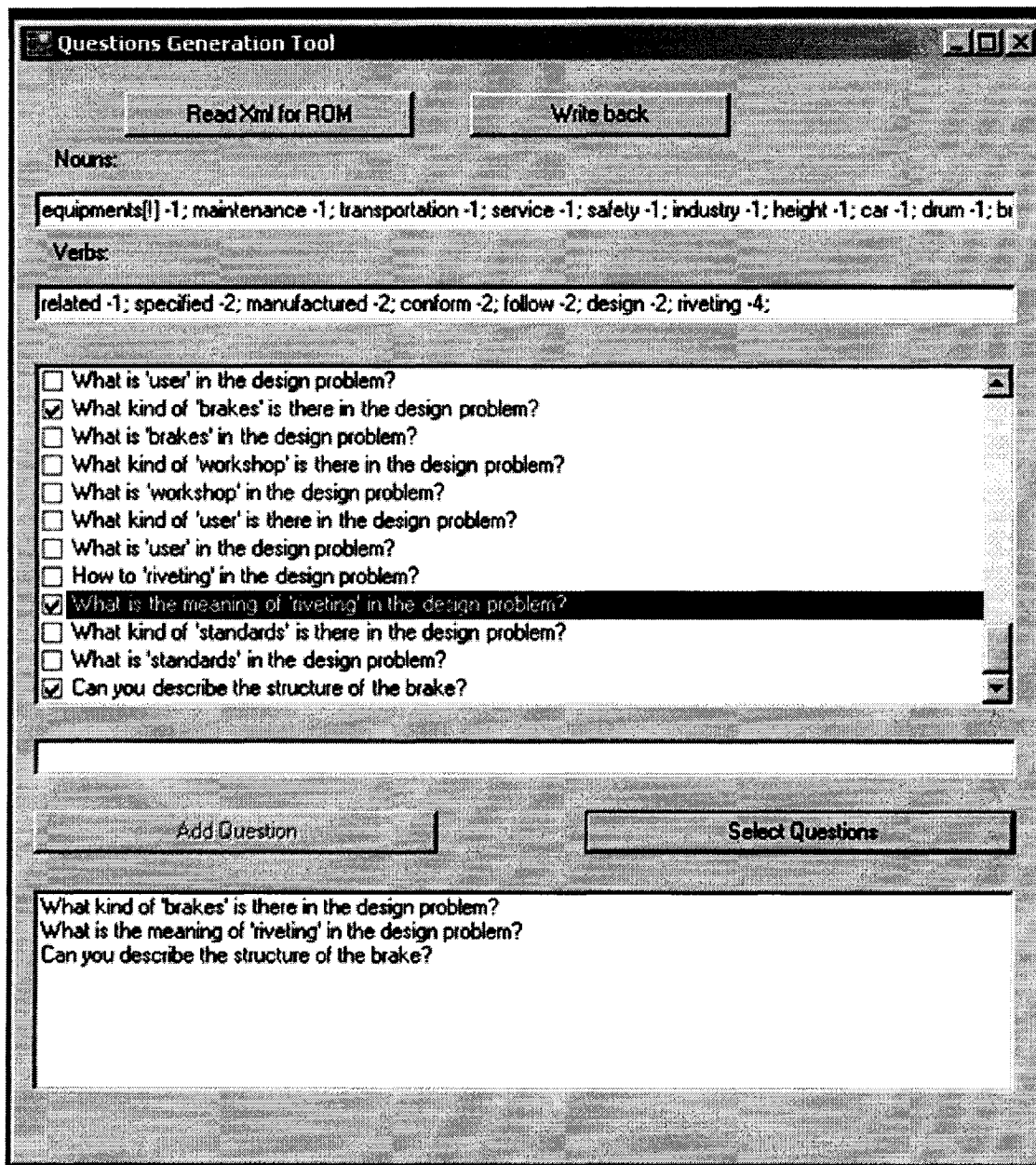


Figure 29 Interface for question generation

Figure 30 shows the interface of question listing. Customer can access the question to answer it. And designer can select the question from the list and check the answer for it.

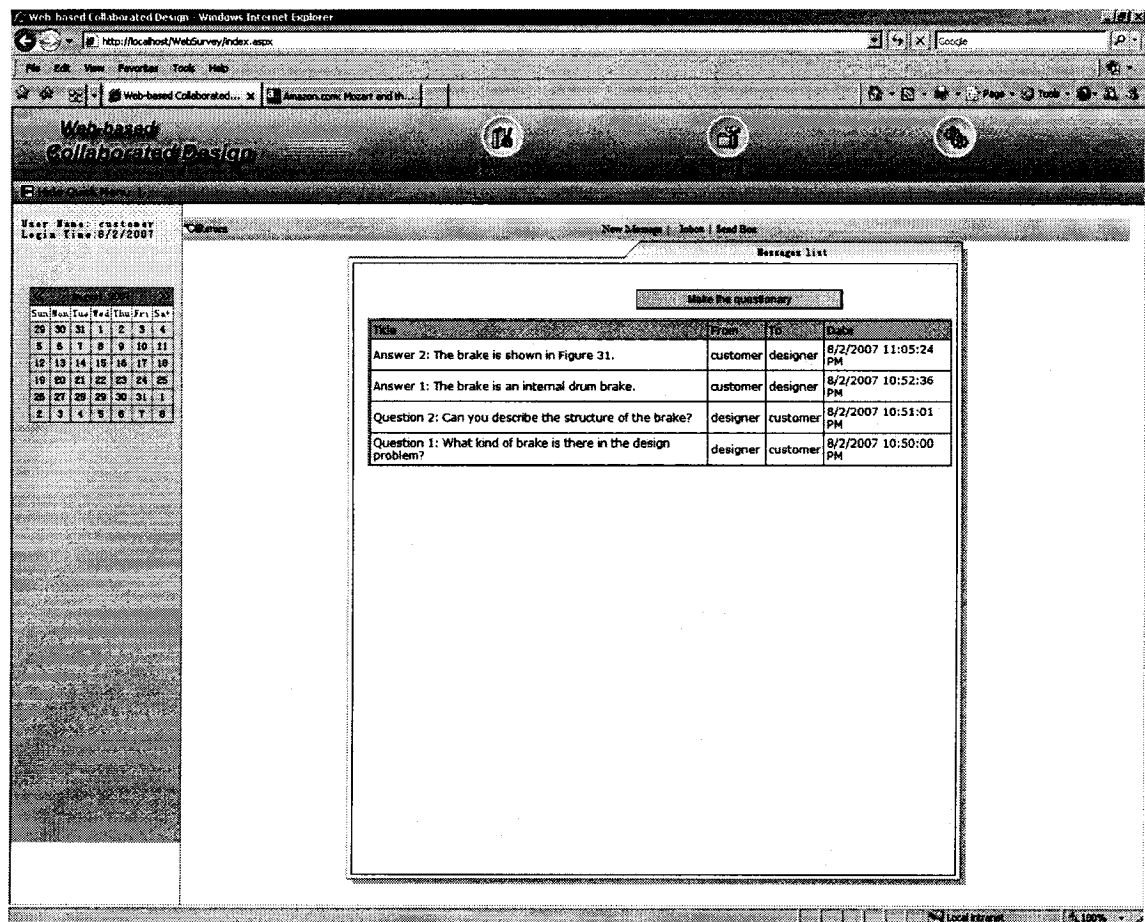


Figure 30 Interface for question listing

5.4 Case study

A rivet setting tool design example is adapted from the book by (Hubka et al., 1988) to illustrate the generic process proposed in this paper and the software prototype. The task of this problem is to design a tool for riveting brake linings onto brake shoes for internal drum brakes as shown in Figure 31.

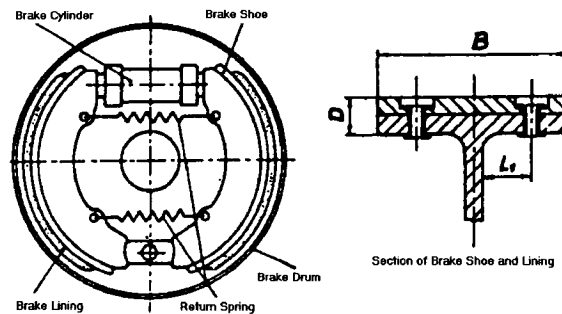


Figure 31 Internal drum brake (Hubka et al., 1988)

In this case study, a customer first presents a design task in natural language as to design a tool for riveting brake linings onto brake shoes for internal drum brakes. We applied the software prototype by following the generic formalization process we provided.

- Step 1: Create ROM diagram.

The ROM diagram based on design problem description is created in the tool of ROMA shown in Figure 32.

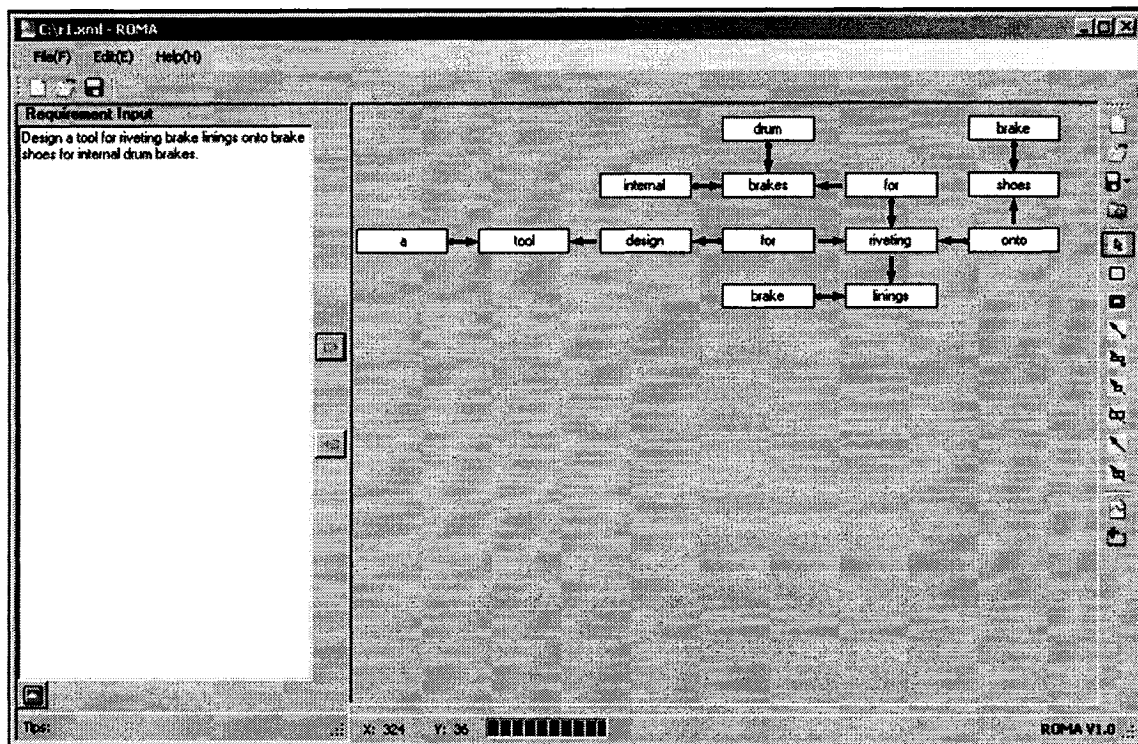


Figure 32 ROM diagram for Step 1

- Step 2: Generate generic questions.

After analyzing the ROM diagram created in Step 1, some questions asked in Step 2 are listed as Table 10.

Table 10 Questions for Step 2

• Q1	What kind of brake is there in the design problem?
• Q2	Can you describe the structure of the brake?

- Step 3: Collect answers.

The answers collected in Step 3 are listed as Table 11.

Table 11 Answers for Step 3

• A1	The brake is an internal drum brake.
• A2	The brake is shown in Figure 31.

- Step 4: Repeat Step1 to Step4 until no more questions can be asked.

Each answer should be analyzed from Step 1 to Step 4 and the ROM diagrams are merged into the original one in Figure 33

Figure 20 Merged ROM diagram for Step 4

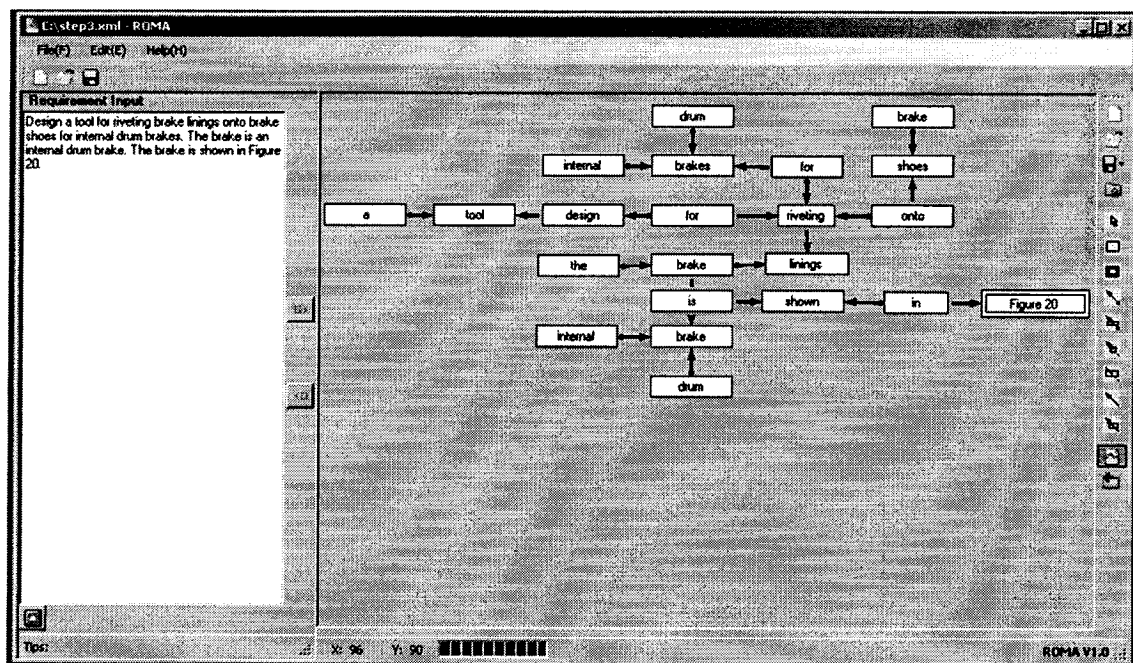


Figure 33 ROM diagram for Step 4

- Step 5: Generate domain specific questions.

In this step, the question of “Q3: What is the life cycle of the tool” is asked first to determine the life cycle the tool is involved in. And the answer to it is collected as “A3: The life cycle includes design, manufacturing, sales, transportation, use and maintenance”.

Based on the classification of requirements in terms of product life cycle of the tool, the questions asked in Step 5 are listed as Table 12.

Table 12 Questions for Step 5

• Q4	What standards should the tool conform to?
• Q5	What standards should be followed for the users?
• Q6	Where the tool should be manufactured?
• Q7	What is the reasonable cost for you?
• Q8	How long will the tool’s service life be?
• Q9	What are the basic functions of the tool?
• Q10	What are the extended functions of the tool?
• Q11	Who will use the tool?

- Step 6: Collect answers to the questions generated in Step 5.

The answers collected in Step 6 are listed as Table 13.

Table 13 Answers for Step 6

• A4	The use of this tool should conform to the related industry safety standards.
• A5	The working height of user should follow ergonomic standards.
• A6	It will be manufactured in a specific workshop, which has specified equipments.
• A7	The cost of this tool cannot be over \$190.00.
• A8	The service life of this tool should be around 5 years.
• A9	The tool rivets brake linings onto brake shoes.
• A10	The tool should be easy for transportation and maintenance.
• A11	The user of this tool is a car mechanic.

- Step 7: Repeat Step 1 to 7 until no more questions can be asked.

The ROM diagrams for the answers of A4 to A11 are listed in the following Figures.

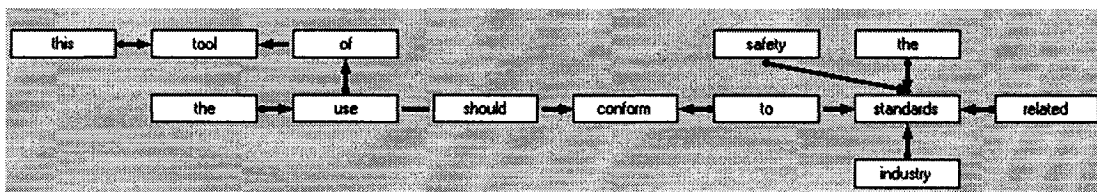


Figure 34 ROM diagram for A4

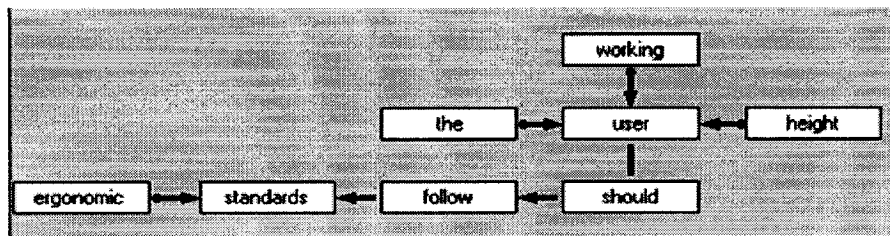


Figure 35 ROM diagram for A5

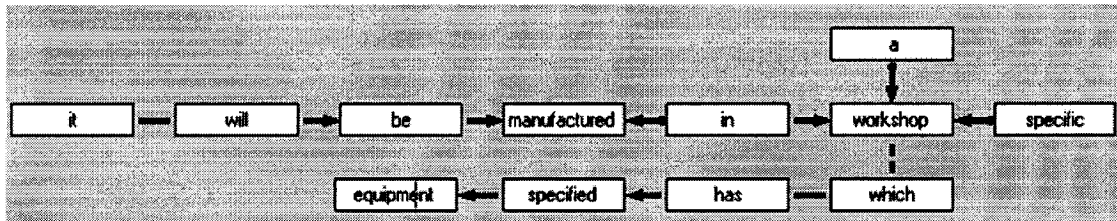


Figure 36 ROM diagram for A6

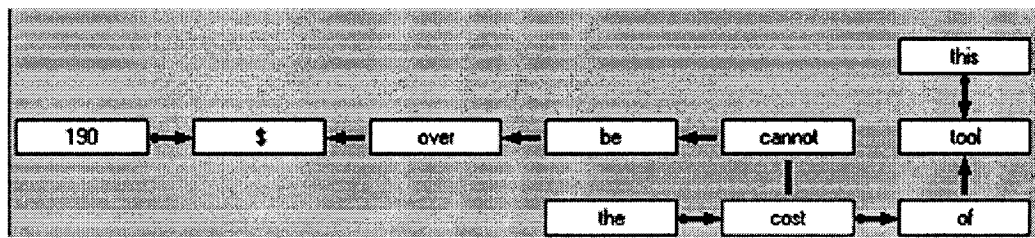


Figure 37 ROM diagram for A7

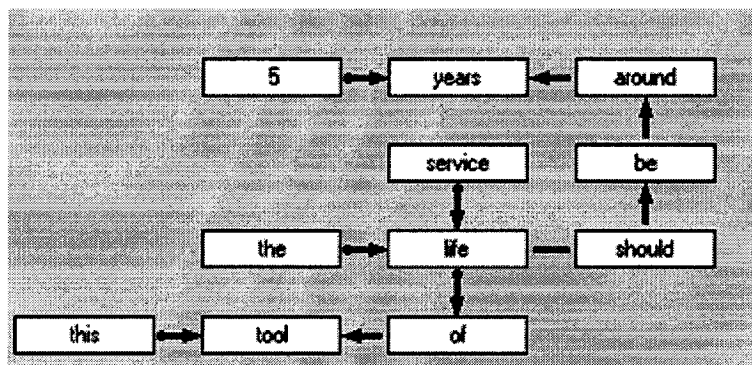


Figure 38 ROM diagram for A8

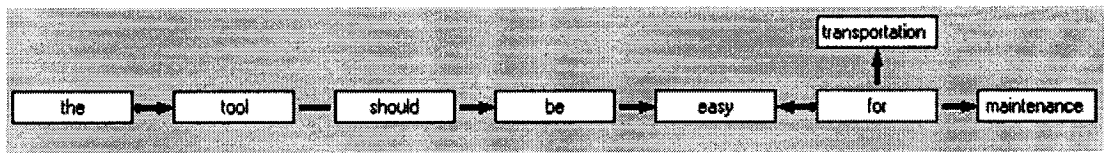


Figure 39 ROM diagram for A10

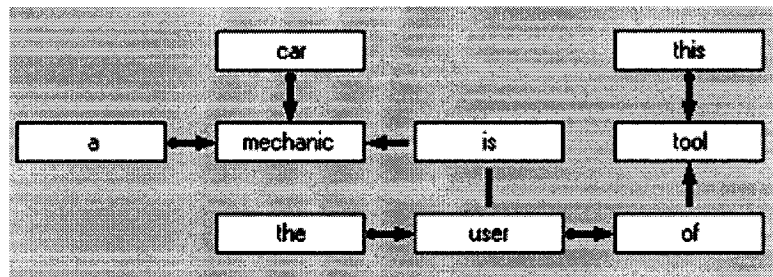


Figure 40 ROM diagram for A11

After merging all the diagrams, the final ROM diagram is shown in Figure 41 and Figure 42.

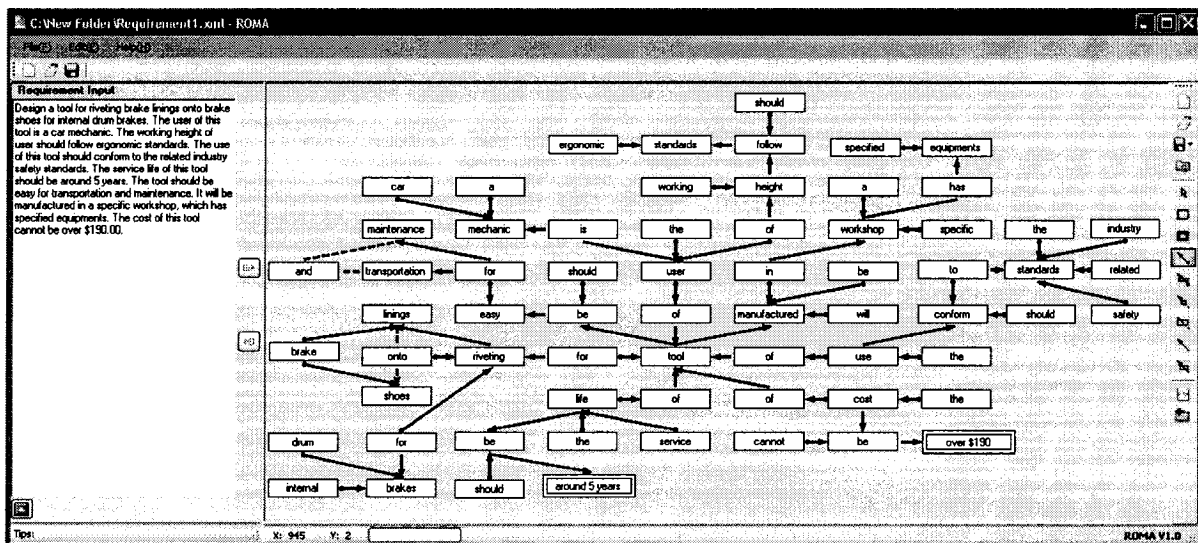
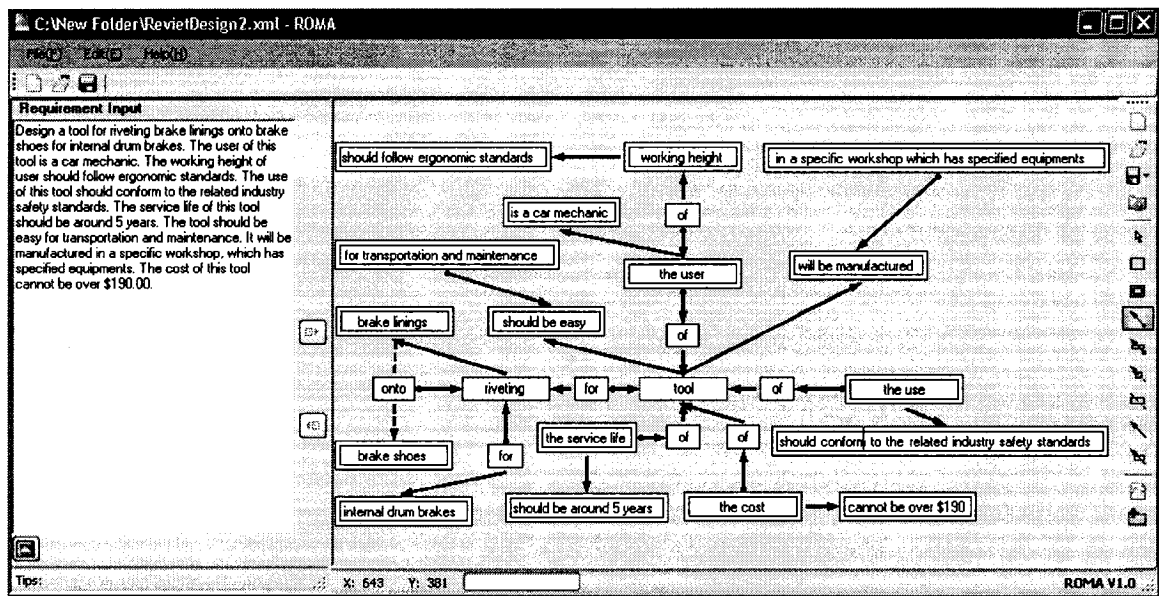


Figure 41 ROM diagram for Step 7



- **Step 8: Output the updated design problem description.**

The final requirements are elicited in the following paragraph.

A rivet setting tool should be designed to rivet brake linings onto brakes shoes. The brake is shown in Figure 31. The user of this tool is a car mechanic. The hand force, the foot force and the working height should follow ergonomic standards. The use of this tool should follow the related industry safety standards. The service life of this tool should be around 5 years. The tool should be easy for transportation and maintenance. It will be manufactured in a specific workshop, which has specified equipments. The cost of this tool cannot be over \$190.00.

Chapter 6

Conclusions and Future Work

6.1 Overview

In this present thesis, a generic process is proposed to elicit precise and complete product requirements. The foundation of this generic process is Environment Based Design (EBD) and Recursive Object Model (ROM), which are derived from the axiomatic theory of design modeling. The EBD indicates that the source of product requirements is product environment and guides designers to generate design solutions by following a recursive process. The ROM provides a graphic and structured representation of linguistic information. Through the ROM analysis of description of design problem using the natural language, the generic process can identify the product to be designed, its environment components, and their relations from the generated ROM diagram by following the rules and templates we provide.

6.2 Conclusions

An iterative inquiry approach is established to elicit precise and complete requirements by asking the right questions. This inquiry approach includes two major algorithms: one for asking generic questions and the other for asking specific domain questions. Both algorithms use predefined rules and templates to generate questions. The generic questions aim to identify the customer's real intent whereas the domain specific questions

aim to collect complete list of requirements. This inquiry approach is an integral part of the environment based design. (Chapter 4)

A software prototype is designed and implemented as a collaboration platform and an analysis tool, which provides effective communications between the design partners. The software prototype helps designers elicit the customer's real intent and collect complete product requirements. A case study about a rivet setting tool design is used to illustrate the concepts proposed in the present thesis and the software prototype (Chapter 5).

6.3 Future work

The initial experiments show that the present approach for requirements elicitation is feasible and promising. However, the inquiry approach needs to be improved through more case studies. Especially, the following issues must be investigated before the approach can be put into industrial applications: the algorithms of asking specific and domain questions, the predefined rules for ROM analysis and product domain analysis, and the question templates for question generation.

To improve the generic process, our research group is approaching it from many perspectives, such as the processing of natural language, especially its ambiguities. The predefined rules for ROM analysis and product domain analysis are being refined. The templates for generating questions are being supplemented. A more robust algorithm for determining the sequence of asking questions is under development. The software prototype can be extended by including a knowledge-base server and an upgraded ROM

tool. At the same time, more cases about software product and others and are being studied.

Publications

1. **Min Wang** and Y. Zeng (2007), Asking the right questions to elicit product requirements, *International Journal of Computer Integrated Manufacturing*, 2007 (Selected from FAIM'07 for inclusion in a special issue, 2007).
2. **Min Wang** and Yong Zeng (2007), Gathering of Product Requirements Based on Linguistic Analysis, *Proceedings of the 17th International Conference on Flexible Automation and Intelligent manufacturing*, Philadelphia, USA, June 18-20, 2007, pp 250-257.
3. Yong Zeng, Lei Chen and **Min Wang**, (2007), Automatic generation and layout of ROM diagram from English text, *in preparation for the declaration of invention to be submitted to the Office of Research*, Concordia University.

References

- Akao, Y. and Glenn, M., 2003. The leading edge in QFD: past, present, and future. *International Journal of Quality and Reliability Management*, 20(1): 20-35.
- Alexander, I., 2003. Misuse cases help to elicit non-functional requirements. *Computing & Control Engineering* 40-45.
- Alvarez, R., 2002. Discourse analysis of requirements and knowledge elicitation interviews, *Proceedings of the 35th Hawaii International Conference on System Sciences*, Big Island.
- Asimow, M., 1962. *Introduction to design*. Prentice Hall.
- Chen, Z., Yao, S., Lin, J., Zeng, Y. and Eberlein, A., 2007. Formalization of product requirements: from natural language descriptions to formal specifications. *Int. J. Manufacturing Research*, 2(3): 362-387.
- Chen, Z.Y. and Zeng, Y., 2006. Classification of product requirements based on product environment. *Concurrent Engineering: Research and Applications, An International Journal*, 14(3): 219-230.
- Christel, M.G. and Kang, K.C., 1992. *Issues in requirements elicitation*, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania.
- Clarke, E. and Wing, J., 1996. Formal methods: state of the art and future directions. *ACM Computing Surveys*, 28(4): 626-643.

- Clausing, D., 1998. Total Quality Development. American Society of Mechanical Engineers, New York.
- Conklin, Jeff and Yakemovic, K.C.B., 1991. A process-oriented approach to design rationale. *Human-Computer Interaction*, 6(3): 357-391.
- Darlington, M.J. and Culley, S.J., 2002. Current research in the engineering design requirement. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 216(3): 375-388.
- Eris, O., 2004. Effective Inquiry for Innovative Engineering Design. Kluwer Academic Publishers, Stanford University, 154 pp.
- Gershenson, J.K. and Stauffer, L., A., 1999. A taxonomy for design requirements from corporate customers. *Research in Engineering Design*, 11(2): 103-115.
- Hickey, A. and Davis, A., 2004. A unified model of requirements elicitation. *Journal of Management Information Systems*, 20(4): 65-84.
- Hubbard, R., Mead, N. and Schroeder, C., 2000. An assessment of the relative efficiency of a facilitator-driven requirements collection process with respect to the conventional interview method, *International Conference on Requirements Engineering*. IEEE Computer Society Press, Los Alamitos, CA.
- Hubka, V., Andreasen, M. and Eder, W., 1988. Practical studies in systematic design. Butterworths.

- Kang, K.C., Cohen, S.G., Hess, J.A., Novack, W.E. and Peterson, A.S., 1990. Feature-oriented domain analysis feasibility study. Software Engineering Institute, Pittsburgh, PA.
- Kantowitz, B., Roediger, H. and Elmes, D., 2001. Experimental Psychology: Understanding Psychological Research. Wadsworth Publishing.
- Kean, L., 1997. Feature-oriented domain analysis, Carnegie Mellon, Software Engineering Institute.
- Knuth, K.H., 2005. Toward question-asking machines: the logic of questions and the inquiry calculus, Tenth International Workshop on Artificial Intelligence and Statistics (AISTATS 2005), Barbados.
- Lecoeuche, R., Mellish, C. and Roberston, D., 1998. A framework for requirements elicitation through mixed-initiative dialogue, International Conference on Requirements Engineering: Putting Requirements Engineering to Practice, pp. 190.
- Lee, D.J. and Thornton, A.C., 1996. The identification and use of key characteristics in the product development process, ASME Design Engineering Technical Conference and Computers in Engineering Conference.
- Leite and Cesar, J., 1987. A survey on requirements analysis, Department of Information and Computer Science, University of California at Irvine.

- Mead, N.R., 2006. Requirements elicitation introduction, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania.
- Meyer, B., 1985. On formalism in specifications. IEEE Software, 2(1): 6-26.
- Minneman, S.L., 1991. The social construction of a technical reality: empirical studies of group engineering design practice. Ph.D. Dissertation Thesis, Stanford University.
- Mullery, G.P., 1979. CORE: a method for controlled requirements specification, Proceedings of the 4th International Conference on Software Engineering (ICSE-4). CA: IEEE Computer Society Press, Munich, Germany, pp. 126-135.
- Nuseibeh, B. and Easterbrook, S., 2000. Requirements engineering: a roadmap, Proceedings of the Conference on The Future of Software Engineering ACM Press Limerick, Ireland pp. 35-46.
- Oxman, R., 2004. Think-maps: teaching design thinking in design education. Design Studies, 25: 63-91.
- Prabhakar, S. and Goel, A.K., 1998. Functional modeling for enabling adaptive design of devices for new environments Artificial Intelligence in Engineering, 12(4): 417-444(28).
- QFD-Institute, 2005. Frequently Asked Questions About QFD. <http://www.qfdi.org/>, [accessed on 5 August 2007].
- Schach, S.R., 2002. Object-Oriented and Classical Software Engineering. McGraw Hill.

- Schiffrin, D., 1994. Approaches to Discourse. Blackwell Publishers Ltd, Oxford, England.
- Simon, H., 1969. The Sciences of the Artificial. MIT Press, Cambridge, MA.
- Southwell, K., James, K., Clarke, B.A., Andrews, B., Ashworth, C., Norris, M. and Patel, V., 1987. Requirements definition and design, The STARTS Guide, Second Edition, . National Computing Centre, pp. 177-313.
- Ullman, D.G., 2002. The Mechanical Design Process. McGraw-Hill, New York.
- Unknown-Author, 1983. IEEE Standard Glossary of Software Engineering Terminology --Institute of Electrical and Electronics Engineers.
- Unknown-Author, 2007. How projects really work. <http://www.projectcartoon.com>, [accessed on August 1, 2007].
- Wang, M. and Zeng, Y., 2007a. Asking the right questions to elicit product requirements. International journal of Computer Integrated manufacturing, Selected from FAIM'07 for inclusion in a special issue.
- Wang, M. and Zeng, Y., 2007b. Gather of Product Requirements Based on Linguistic Analysis, Proceedings of the 17th International Conference on Flexible Automation and Intelligent manufacturing, Philadelphia, USA, pp. 250-257.
- Wood, J. and Silver, D., 1995. Joint Application Development. Wiley, New York.

- Wootton, A.B., Cooper, R. and Bruce, M., 1998. Requirements capture: theory and practice. The Engineering and Physical Sciences Research Council Technology Management Initiative, 18(8-9): 497-511
- Zahniser, R.A., 1990. How to Speed Development with Group Sessions. IEEE Software: 109-110.
- Zeng, Y., 2002. Axiomatic theory of design modeling. Transaction of SDPS: Journal of Integrated Design and Process Science, 6(3): 1-28.
- Zeng, Y., 2003. Formalization of design requirements, Integrated Design and Process Technologies, IDPT-2003, Austin, Texas, December 3-6.
- Zeng, Y., 2004a. Environment-based design: process model, Concordia Institute for Information Systems Engineering, Concordia University, Montreal.
- Zeng, Y., 2004b. Environment-based formulation of design problem. Transaction of SDPS: Journal of Integrated Design and Process Science, 8(4): 45-63.
- Zeng, Y., 2007. Recursive Object Model (ROM) - Modeling of Linguistic Information in Engineering Design. Computers in Industry (*submitted*).
- Zeng, Y., Chen, L. and Wang, M., 2007. Automatic generation and layout of ROM diagram from English text. In: D.t.C. University (Editor), Patent Application, Canada.
- Zeng, Y. and Cheng, G.D., 1991. On the logic of design. Design Studies, 12(3): 137-141.

- Zeng, Y. and Gu, P., 1999a. A science-based approach to product design theory Part I: Formulation and formalization of design process. *Robotics and Computer-Integrated Manufacturing*, 15(4): 331-339.
- Zeng, Y. and Gu, P., 1999b. A science-based approach to product design theory Part II: Formulation of design requirements and products. *Robotics and Computer Integrated Manufacturing*, 14(4): 341-352.
- Zeng, Y., Pardasani, A., Antunes, H., Li, Z., Dickinson, J., Gupta, V. and Baulier, D., 2004. Mathematical foundation for modeling conceptual design sketches. *Transactions of the ASME: Journal of Computing and Information Science in Engineering*, 4(2): 150-159.
- Zeroual, K., 1989. An Approach for Automating the Specification-Acquisition Process, In *Proceedings of the Second International Workshop on Software Engineering and Its Applications*, Toulouse, Nanterre, France, pp. 349-355.