# Knowledge-Based Inventive Conceptual Design

A thesis submitted to the
University of Wales, Cardiff
for the degree of

**Doctor of Philosophy**

by

**Huimin Liu**

Manufacturing Engineering Centre
School of Engineering
Cardiff University
United Kingdom

**2007**

# Abstract

Conceptual design is the first phase of the design process. Most basic functions of a new product and the solutions for solving design problems are generated in this critical phase, which will affect the attributes in the later detailed design process. Conceptual design, especially the process of concept generation, is an innovation process that is achieved by human intelligence. The intuition and experience of designers play a significant role during the design process which is hard to be replaced by computer-aided tools or artificial intelligence technology.

TRIZ is an inventive problem-solving tool to help people improve creativity. It is applied in this work to generate creative design concepts. The TRIZ inventive principles are extended by integrating other TRIZ tools and TRIZ-derived tools. These principles are also restructured by the inspiration of I-Ching. The Behaviour-Entity representation of inventive principles enables the generation of new and innovative solutions based on TRIZ.

The TRIZ Contradiction Matrix (CM) and inventive principles are then used to develop the TRIZ-based concept generation approach by adding constraints to the standard Behaviour-Entity representation of TRIZ. This approach is developed to retrieve modified TRIZ inventive principles and to generate new solutions by re-organising the BEC (Behaviour-Entity-Constraint) representation of principles according to the conflicting design requirements.

Finally, a negotiation-based approach is integrated with an existing no-compromise approach to develop a knowledge-based system for automatically detecting and resolving conflicts. The recommendation is given as an output arranged by weight to help the designer improve creativity and efficiency for concept generation and conflict resolution in conceptual design. The approach is implemented by using a rule-based language, JESS. A case study of aircraft fuselage layout design is presented to demonstrate the benefits of using this conflict resolution system.

## Acknowledgements

I would like to thank my supervisor Prof. D. T. Pham for his excellent supervision, continuous encouragement and support throughout my work. Without his support I could not have reached this far.

I also want to thank all members of the Design Group and Intelligent Systems Laboratory, who were very good to me and very helpful whenever I needed them.

Finally, my deepest gratitude is to my parents for their continuous support and understanding.

# Contents

# List of Figures

# List of Tables

## Abbreviations

| | |
|---|---|
| AI | Artificial Intelligence |
| APIs | Application Programming Interfaces |
| BDN | Building Design Network |
| BEC | Behaviour-Entity-Constraint |
| CAD | Computer-Aided Design |
| CE | Concurrent Engineering |
| CKBS | Cooperative Knowledge-Based Systems |
| CM | Contradiction Matrix |
| FAST | Functional analysis systematic technique |
| FBS | Function-Behaviour-State |
| NLP | Neuro-Linguistic Programming |
| IFR | Ideal Final Result |
| IPM | Inventive Principles Matrix |
| JESS | Java Expert System Shell |
| KBS | Knowledge-Based System |
| USIT | Unified Structured Innovative Thinking |

# Chapter 1. Introduction

## 1.1 Overview

This chapter briefly introduces the research presented in this thesis. The motivation of this research is discussed. Next, the objectives of the research are outlined. Finally, the layout of the remaining structure of the thesis is given.

## 1.2 Motivation

Product development is concerned with the design, manufacture, assembly, distribution, marketing of products, and so on. Over the past few years, it has been a focus of research for both academics and industrialists. This is due to the widespread recognition that competitive advantage can be achieved by effective product development in industry.

Engineering design plays a crucial role in the process of new product development. It is concerned with the development of detailed specifications for a product which provide a technical function. In order to increase the effectiveness of design, several computer technologies have been developed to support design, known as Computer-Aided Design (CAD).

Conceptual design is the first phase of the design process. Most basic functions of a new product and the solutions for solving design problems are generated in this critical phase, which will affect the attributes in the later detailed design process. Artificial intelligence (AI) and computer technologies have been integrated to provide computer support for automated design which CAD fails to address.

However, conceptual design, especially the process of concept generation, is an innovation process that is achieved by human intelligence. The intuition and experience of designers play a significant role during the design process which is hard to be replaced by computer-aided tools or artificial intelligence technology.

At the same time, design is a demanding process that requires expertise in many different fields such as science, engineering, and often art, all of which are distributed in different phases. Most of the literature assumes that conflict exists commonly in engineering design. However, there are few computational tools to provide support for the human designer to detect and resolve conflict in the early stages of design.

In most cases, creative activities such as generating new solutions or detecting and resolving conflicts are still left to human experts. There are a variety of techniques to help people improve creativity, such as morphological analysis, brainstorming, lateral thinking, TRIZ etc. Compared with other creative thinking tools, TRIZ is one of the most powerful tools as it provides not only a general method for breaking out of the patterned way of thinking, but also a series of tools for solving technical problems.

Among the many TRIZ tools, the TRIZ contradiction matrix and 40 inventive principles are capable of coping with both of the issues in conceptual design identified above, which are innovative concept generation and conflict resolution.

However, TRIZ inventive principles have limitations such as their often illogical sequencing, their level of overlap and the gaps that they contain. As I-Ching and TRIZ use the same philosophy of dialectics thinking. The principles can be restructured and improved by the inspiration of I-Ching.

These modified TRIZ tools involve massive knowledge for resolving innovative problems. They can be integrated with the AI technique of Knowledge-Based Systems (KBS) to improve automation, efficiency, and creativity in conceptual design.

The starting points for the investigation of the proposed method are research questions and hypotheses, which can be summarised as:

1. I-Ching can be a suitable tool for modifying TRIZ inventive principles.
2. Modified TRIZ-based design methodology can be more efficient than classical TRIZ.
3. Integration of TRIZ and negotiation-based methods can provide inventive conflict resolution strategies.

The methodologies used in this research to prove the above hypotheses involve classifying TRIZ inventive principles, analysing symbolic expressions, developing a TRIZ-based design model, including a mathematical model of conflict resolution, and conducting a case study of aircraft fuselage design. The details of the research methods are described in Chapter 3, Chapter 4, and Chapter 5 of this thesis.

## 1.3 Objectives

The main objectives of this research are therefore as follows:

- To evolve TRIZ inventive principles and apply them to supporting creative concept generation.
- To integrate a negotiation-based approach and TRIZ to support conflict resolution.
- To provide, using KBS, computer support for this concept generation and conflict resolution at the stage of conceptual design.

## 1.4 Outline of the thesis

This thesis comprises six chapters. The remainder of its structure is as follows:

**Chapter 2** reviews the background literature relevant to the work presented in the thesis.

**Chapter 3** discusses the representation of TRIZ inventive principles as symbolic expressions. It also presents a collection of modified principles based on this representation.

This chapter discusses how 40 TRIZ inventive principles are restructured by analysing the similarities between them and I-Ching. Then the symbolic expression for representing an inventive design solution is defined based on the new structure. The main meaning of the principles and other TRIZ derived tools are extracted and represented as symbolic expressions. They are then analysed and compared to remove redundant information and the new principles are selected.

**Chapter 4** describes an approach that employs a knowledge base comprising rules that implement a Behaviour-Entity-Constraint (BEC) representation of modified TRIZ inventive principles.

A TRIZ-based concept generation model is presented. A case study conducting the design of the passenger cabin layout in an aircraft by applying the proposed method and a method using classical TRIZ is described to show the benefits of the new TRIZ-based approach.

**Chapter 5** presents the detection and resolution of three types of conflict in conceptual design. It also details the implementation of a rule-based system developed using JESS (Java Expert System Shell) to generate automatically design concepts and the resolution of conflicts.

This chapter proposes an inventive conflict resolution model. A mathematical model for calculating the weights of conflict resolution strategies is described. The example of aircraft fuselage is used to illustrate the application of the method.

**Chapter 6** presents the conclusions, contributions of the research and recommendations for further study.

# Chapter 2. Literature review

## 2.1 Overview

This chapter reviews the background literature relevant to the work presented in this thesis. First, conceptual design as an essential phase of engineering design is surveyed. This is followed by a review of current techniques supporting four phases of conceptual design. These four phases are: design specification and representation, concept generation, concept selection, and evaluation. Next, the literature on conflict resolution in conceptual design is reviewed. TRIZ, an inventive problem-solving tool, has been applied to solving engineering design conflicts and to generating inventive design concepts. This chapter then reviews the literature on TRIZ and its application to conceptual design.

## 2.2 Conceptual Design

Many researchers have tried to present a definition of conceptual design relevant to their own research background. However, so far there is not a well-accepted definition for conceptual design. According to previous literature articles, conceptual design has the following common features.

- New and innovative: the conceptual design is a very important task in computer-aided-design (CAD) [Wang 1994], particularly when designing new and innovative products, or when generating a completely new design for an existing product [Wang 2002].

- Difficult: knowledge of the design requirements and constraints during this early phase of a product's life cycle is usually imprecise and incomplete, making it difficult to utilise computer-based systems or prototypes [Hsu 2000]. It involves the formulation of abstract ideas with approximate concrete representations [Takala 1989].
- Abstract: the conceptual level is an abstraction in which a system is specified incompletely [Lidsky 1998]. Conceptual design commences with high-level descriptions of requirements and proceeds with a high level description of a solution [McNeil 1998].
- Integrated: it is the phase where engineering science, practical knowledge, production methods, and commercial aspects need to be brought together [French 1985].

Therefore, conceptual design is perhaps the most crucial task in an engineering product development cycle [Wang 2002]. It is the phase that makes the greatest demands on the designer, where there is the most scope for significant improvements, and where the most important decisions are taken [French 1985].

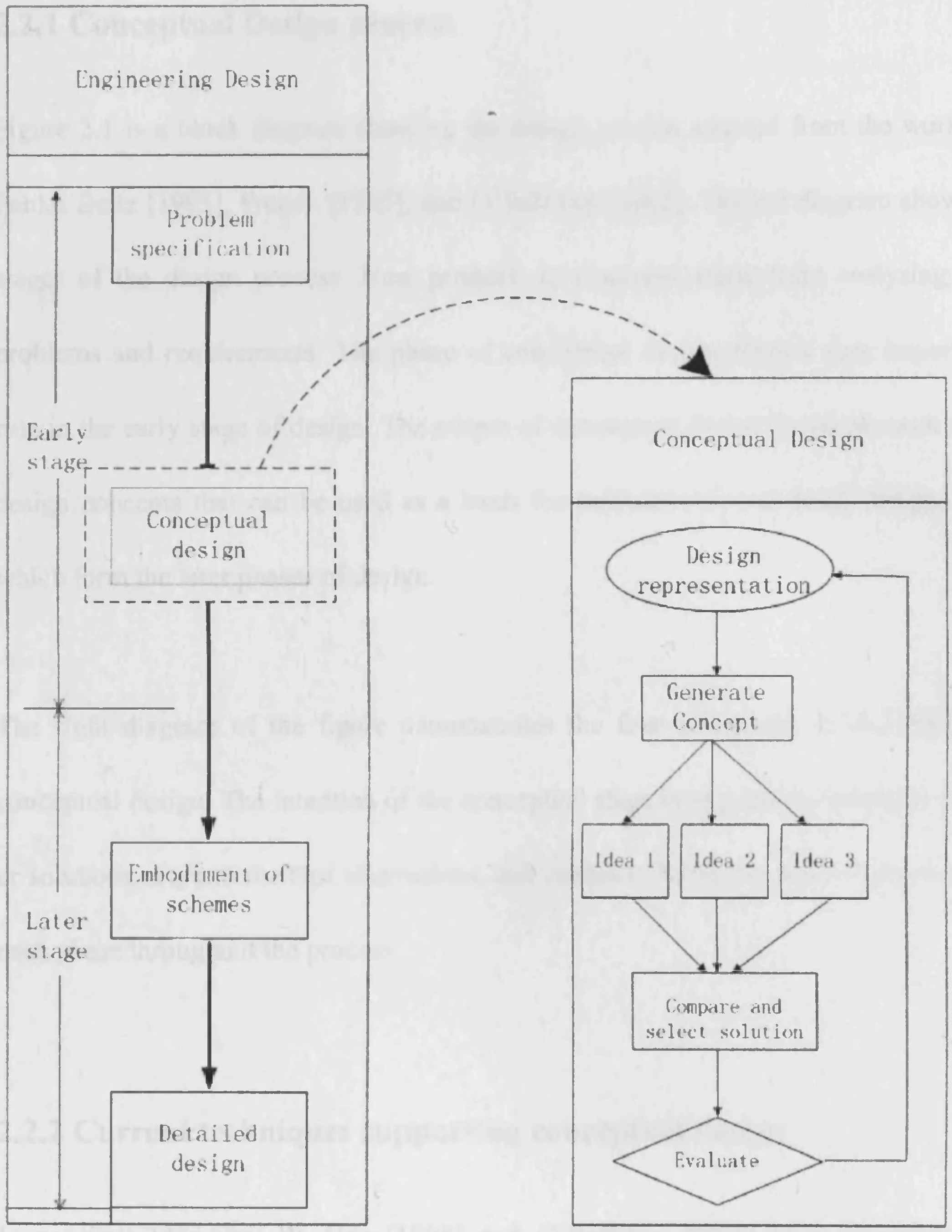


Figure 2.1 Process of design and conceptual design

### 2.2.1 Conceptual Design process

Figure 2.1 is a block diagram showing the design process adapted from the work of Pahl& Beitz [1995], French [1985], and O'Sullivan [2002]. The left diagram shows 4 stages of the design process. New product development starts from analysing the problems and requirements. The phase of conceptual design plays a very important role in the early stage of design. The output of conceptual design is one or more new design concepts that can be used as a basis for embodiment and detail design, and which form the later phases of design.

The right diagram of the figure demonstrates the four sub-stages in the phase of conceptual design. The intention of the conceptual stage is to generate inventive ideas or solutions, explore the best alternatives, and evaluate the output that is generated in each phase throughout the process.

### 2.2.2 Current techniques supporting conceptual design

In previous literature, W. Hsu [1998] and O'Sullivan [2002] have classified the techniques for supporting the conceptual design stage into two groups: modelling issues and reasoning issues. The difficulties of the first category occur usually at the early phase of conceptual design while the reasoning techniques are used for supporting concept generation, selection, and evaluation.

Pahl& Beitz [1995] presented a diagram showing detailed steps in conceptual design which can be mapped into four general phases of conceptual design as outlined above. In the design literature there are many tools or techniques for supporting these stages. Figure 2.2 shows the relationships between techniques and corresponding stages.

#### 2.2.2.1 Design representation

Conceptual design starts from establishing the function structure. A function block diagram [Pahl& Beitz 1995] is used to describe overall function based on the flow of energy, material, and signals and to express the relationship between inputs and outputs. Functional Analysis Systematic Technique (FAST) [Miles 1965] diagrams are used to form a hierarchical tree to represent functional relationships. Functional flow charts and functional logic diagrams are other examples. [Sturges 1993]. Campbell [1999] developed functional representation based on qualitative physics [Forbus 1988], bond graphs [Paynter 1961; Ulrich 1989], and functional block diagrams [Pahl 1996].

Representation of design modelling is also initiated at this first stage of conceptual design. There are a variety of design models for representing different aspects of design knowledge. The main modelling techniques are the Grammar-based model, Geometry-based model, Graph-based model, and Knowledge-based model.

#### Grammar-based

A language is defined by its grammar. It expresses the engineer's intention of understanding and formalising a design explicitly. A few related researches have been reported in the literature, such as CFRL [Vescovi 1993], and CANDLE [Andersson 1995] etc. CFRL is a formalism for representing both device functions and behavioural knowledge. CANDLE is a modelling language that enables the use of engineering terminology for modelling early design work. However, these languages are used to describe only some aspects of design for specific domains.

A domain-independent ontology, YMIR, is proposed by Alberts [Alberts 1994]. Alberts discusses the use of a shareable ontology for the formal representation of engineering design knowledge in different design domains.

Languages are computationally efficient and unambiguous and they are a powerful way of structuring knowledge. However, they fail to account for the relation between shape and function and they are not able to represent the complex reasoning demanded by the conceptual design activity.

**Geometry-based**

Geometry-based representation focuses on representing the shape of a product. Existing representations of geometric shapes in literature are B-rep, CSG, and variational modelling [Finger 1989]. Another type of geometry-based representation is feature-based modelling. As features provide function and are generally described geometrically, the use of features as carriers of function in design has been an important area of research for a number of years. Brunetti [2000] introduces an approach towards a feature-based integrated product model that incorporates a feature-based representation scheme for capturing the product semantics handled in the conceptual design phase and links early design with part and assembly modelling. EDISON [Dyer 1986] is an example of a system using feature-based modelling. It has a database of known mechanisms and is indexed by their functions, structures, and the situations in which they are used.

Geometry models allow designers to represent physical forms in computer format. However, they cannot represent the intricate knowledge in processing and selecting one physical form over another.

### Graph-based

Graph representation is widely used in the conceptual design stage. It has been used to model all aspects of a product's function, behaviour and structure. A Function-Behaviour-State (FBS) diagram [Umeda 1996] is used to represent a design object hierarchically. The FBS diagram represents a function as an association of function symbols and behaviours, rather than just one of them.

One main advantage of using graphs to model different aspects of design is that graph theory is a developed field of study. Al-Hakim [2000] used graph theory to represent a product and the relationships between its components. This representation provides a more refined visualisation of the energy flow and is applicable to numerous designs.

**Knowledge-based**

Complete design knowledge of requirements and constraints is difficult to obtain at the early stage of engineering design. This highly skilled task is very complex and requires different types of knowledge models to facilitate reasoning. The rule-based paradigm has been adopted by Li [1996]. to automate the computational synthesis of the conceptual design of mechanisms. Besides rule representation, frame representation is also widely used. In the paper by Tong and Gomory [1993], they used a frame-based structure to model parts of standard kitchen appliances. Navinchandra [1991] and Hsu proposed case representations to support conceptual design activities. Besides the theory part in literature, one of the significant knowledge-based models for conceptual design is Scheme-builder [Bracewell 2002].

Knowledge models aim to capture the human designer's reasoning process. Yet, they are unable to fully model the real world constraints and to simulate commonsense reasoning.

Apart from the representation forms discussed above, other modelling techniques have been used in conceptual design, such as object-based and image-based modelling. Object-oriented techniques [Marefat 1993] provide the modelling flexibility needed for conceptual design. Visual thinking had its beginning in 1969. McKim [1980] demonstrated through experimental studies that visual thinking is vital to all branches of design practice.

It can be seen that each existing modelling technique has its strengths and weaknesses. Complex knowledge in conceptual design cannot be captured by any individual representation. The trend has been towards the integration of various representation schemes. In spite of the tremendous efforts in deriving a suitable modelling technique for conceptual design, there is still a long way to go. The issue of incomplete and abstract knowledge has not been addressed fully [Hsu 2002].

### 2.2.2.2 Concept generation

Concept generation is the most critical phase in conceptual design. Many tools and techniques in the literature have focused on this area. In general, the techniques can be divided into two groups: human-oriented creative thinking tools and the computer-aided approaches. Human-oriented tools focus on assisting designers to generate ideas, while computer-aided approaches concentrate on generating concepts automatically.

**Creative thinking tools**

Concept generation is where engineers use creativity and imagination to develop approaches to achieve the design objectives while satisfying the constraints [Hyman 1998].

There are a variety of techniques to help people to generate new design ideas, such as

Morphological Analysis, Brainstorming, Lateral Thinking, Attribute Listing, Checklisting, Synectics etc [Wei 2000, Proctor 1997]. This large collection of techniques can be classified into two groups: disciplined thinking and 'out-of-the-box' or divergent thinking. Disciplined thinking relies on logical or structured ways of creating a new product or service. Examples of this approach are Morphological Analysis and Reframing. One of the well-known 'out of the box' thinking methods is lateral thinking [Bono 1970], which can be used to break out of patterned ways of thinking.

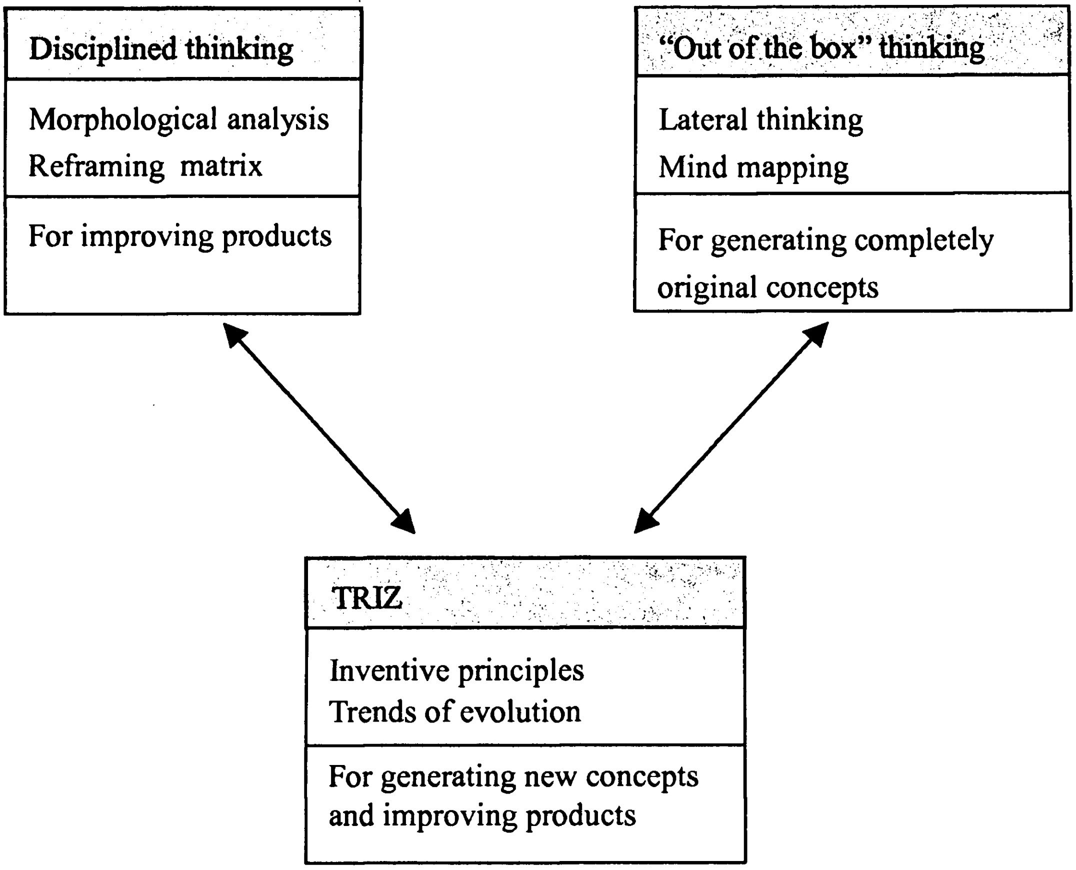


Figure 2.3 Comparison of creative thinking tools

However, each type of approach has its strengths and weaknesses (Figure 2.3). Logical, disciplined thinking is very effective in making products and services better. However, it can only achieve so much before all practical improvements have been carried out. Divergent thinking can generate completely new concepts and ideas, and create exceptional improvements to existing systems. In the wrong place, however, it can be sterile or unnecessarily disruptive.

TRIZ [Altshuller 1988] is the acronym for "Theory of Inventive Problem Solving," in Russian. TRIZ integrates the advantages of disciplined and divergent thinking. For example, a contradiction matrix (CM) provides a structured way of solving technical problems while the Ideal Final Result (IFR) visioning method is a divergent thinking technique based on the philosophy of "breaking psychological inertia". Although TRIZ provides a number of tools for helping people generate new ideas, it has its limits. For example, tools encapsulated in TRIZ are difficult for people to understand and apply in practice. However, compared to other creative thinking tools, TRIZ is still one of the most powerful tools for generating ideas during conceptual design.

### Computer-aided approach

Human-oriented approaches can help the designer generate creative ideas, while computer-aided approaches can provide the designer with various levels of support at the stage of concept generation. Reasoning techniques are the core of computer-aided design systems. According to Pahl& Beitz, generating concepts involves several tasks. The main issues on reasoning techniques for supporting these tasks are: searching working principles, retrieval, combination, and optimisation of solutions.

Search techniques such as genetic algorithms, evolutionary search, simulated annealing, and hill-climbing have been used as a basis for searching working principles in conceptual design. Genetic Algorithms [David 1989] have been widely used to generate candidate solutions [Michael 1995; Ian 1994; Bentley 1996; Bentley 1997]. The use of GAs for generating design solutions automatically is often integrated with other techniques such as backtracking, constraint propagation, and Neural Network [BOS. 1998]. Rafiq et al [1999] developed a concept generation tool using genetic algorithms integrated with neural networks. The system produces a single optimal design although information regarding other options produced by the GA can be accessed by the user. The Qualitative Reasoning technique [Li 1996] has been integrated with a heuristic approach to generate feasible design alternatives.

Knowledge bases are used to capture and store product and procedural design knowledge. A number of knowledge-based reasoning systems for supporting retrieval of solutions have been reported in the literature [Kawakami 1996; Sudhakar 1996; Sabouni 1997; Sun 1994]. Case-base is one type of knowledge-based system. Case-based reasoning is a relatively new approach that applies past experience stored in a computerised form towards solving problems in similar contexts. CADET

[Sycara 1991] is a successful case-based design system that retrieves previous successful designs while avoiding previous failures such as poor materials or high costs.

One of the challenges for supporting the combination of working principles for a suitable solution is combinational explosion. Carmell [1998] proposed an approach that involves evaluating the individual solutions to identify the optimum ones and then combining only those to obtain an overall solution. Li [2000] presented a filtered strategy to reduce the extent of the combination explosion.

Various optimisation techniques have been applied for finding optimal solutions. The Pareto optimality concept [Vancza, 1999], Operations research methods, and adaptive methods are the focus of the research in this area. Koski [1990] presented a survey of the state-of-the art optimisation techniques with a focus based primarily on the Pareto optimality concept. Recently, a number of researchers have begun to apply the principle of Pareto optimality to a wide variety of problems in design. [Gero 1995; O'Sullivan 1998; Bowen 1998; Charles 1995]. Levary [1988] draws attention to the interaction between operation research techniques and engineering design. The adaptive methods have been used to improve the features of existing designs through the optimisation of variables defined within the system.

In general, concept generation is the core phase of conceptual design and several computer-aided approaches have been applied for supporting the designer to accomplish different tasks automatically at this stage. As mentioned above, TRIZ provides a collection of innovative knowledge to generate design concepts. Thus, knowledge-based reasoning can be integrated with TRIZ to automate the retrieval of TRIZ-based design solutions.

### 2.2.2.3 Concept selection

The phase of concept selection involves several tasks: the selection of suitable combinations and the firming-up into principle solution variants. The research in this area also focuses on applying a computer aided approach to support the designer. Sieger and Salmi [1997] proposed an expert system approach to guide the selection and coupling processes. Fruchter [1991] applies qualitative reasoning at different structural abstraction levels to select design modifications arising from performance problems of lateral load resisting frame structures. Approaches developed to support concept selection are generally based on the techniques used in concept generation. Koski [1990] classified the multi-criteria structural design process into three phases. The first phase is the problem formulation, where the criteria, constraints, and design variables are chosen. The second phase is the generation of Pareto optimal solutions. The final phase describes the decision-making procedure employed to select the best compromise solution.

#### 2.2.2.4 Evaluation and verification

Evaluation and verification is the last phase of conceptual design in the textbooks. However, it should be applied during each phase of the early design process. [Huang 1999] proposed a web-based morphological concept assessor to assist designers to perform the evaluation by narrowing down the feasible alterative concepts based on a morphological evaluation chart. Reasoning techniques have also been used in this area. A constraint-based approach was proposed by Deng [2000] to perform functional design verification automatically. Simulation is an important approach for evaluation. [Sieger 1997] measured design performance through simulation. The simulation engine utilises discrete event system specification as its formalism.

## 2.3 Conflict resolution

As discussed above, concept generation is a crucial phase that relies on the engineer's creativity and experience in conceptual design. Similarly, conflict resolution also requires creative negotiation activities that can be difficult to be replaced by any AI or computer-aided tools. Most of the literature assumes that conflict exists commonly in engineering design [Klein 1991, Lander 1997]. Different definitions are given for the concept of conflict [Slimani et al. 2004]. In the product design domain, Cointe [1998] defines a conflict between designers as a disagreement about the product components or their evolution. According to Klein [2000], a conflict is an incompatibility between two (or more) decisions concerning the design or the designer's goals. It has been stated that a conflict is linked to a divergence of goals [Castelfranchi 2000].

Design can be considered as a decision-making process. An individual designer needs to specify multiple design goals and make a number of decisions during this process. Conflicts will occur if these goals or decisions are not compatible. In the concurrent engineering (CE) environment, conflict becomes much more explicit in that all members involved in a design effort are expected to join in the discussion and negotiation over any disagreements encountered [Owen 1986]. A review of conflict resolution approaches in conceptual design is given below.

### 2.3.1 Negotiation approach

Early work on conflict study focused on conflict in group interaction and human participants' behavior. [Slimani 2006]. A good number of computer-aided models of negotiation or compromise have been developed. Werkman [1990] presented a knowledge-based model of negotiation that includes an arbitrator in the negotiation process to store the agent's perspective and to collect the inter-agent dependencies. Sycara [1991] proposed a case-based and utility negotiation to arrive at a compromise solution. CONCENSUS [Cooper 1998] is a multi-agent system for conflict negotiation between humans and agents. Similarly, an agent-based system (cooperative expert systems, domain independent, general purpose, object-oriented, built in Prolog) for conflict resolution was developed by Huang [1993]. Negotiation Lens [Cedrone 2004] is a tool to facilitate conflict negotiation intended to produce benefit for all human designers. The good negotiation results are obtained by moving the diverse groups away from an adversarial stance and back into a collaborative relationship.

It is found that negotiation is a widely accepted approach for conflict resolution in collaborative design. Moreover, existing developments are based on single approaches (e.g. negotiation) to conflict resolution. It is also the case that no development uses conflict modeling and analysis to attempt an early, cost effective conflict resolution. [Lara 1999]

### 2.3.2 Multi-stage approach

The process of compromise/negotiation has serious drawbacks [Labovitz 1980]. Labovitz also reports studies that show that effective conflict resolution can be achieved by both confrontation of disputing parties and by case settlement by an arbitrator, rather than by compromise/negotiation.

PERSUADER, a general negotiation model that solves labour-management disputes was presented by Sycara [1990]. PERSUADER uses case-based reasoning to resolve multi-agent conflicts, and uses a mediation agent to select a solution when case-based reasoning cannot be applied. Wong [1997] introduced BDN (Building Design Network), a computer program that supported a conflict resolution mechanism. The mechanism is composed of four stages: inquiry, arbitration, persuasion, and accommodation. These methods can be combined in an order appropriate to the application domain such that if one method fails, the system will try the next. A Concurrent Engineering Negotiation Support System (CONCENSUS) was introduced by Cooper et al. [1998]. A high-level control mechanism for multi-party negotiation was described. Cooper's mechanism is the same as that of PERSUADER and BDN in that if a conflict resolution approach fails to solve the conflict situation, another is instantiated. This multi-stage mechanism improves the efficiency of conflict resolution and will be integrated into the proposed approach.

In general, *compromise* or *tradeoff* is still the main approach for resolving conflicts in engineering design. However, the two sides of a conflict can only be partially satisfied by compromising. Therefore, the approach of solving conflicts without compromise, TRIZ [Altshuller 1988], is considered for supporting a more creative and effective conflict resolution, which will be introduced and reviewed in next section.

## 2.4 TRIZ

### 2.4.1 TRIZ inventive principles

TRIZ was developed by G. Altshuller and his colleagues in the former USSR starting in 1946, and is now being practiced throughout the world. Besides the original TRIZ theory, some researchers have presented TRIZ-based systematic thinking methods by combining their experience and latest research work [Salamatov 1999; Mann 2002].

The set of 40 Inventive Principles of classical TRIZ were extracted and derived from thousands of patents in the Russian patent databases. It is a useful creativity tool for a variety of problem solving situations [Fey 2005]. However, the principles are frequently criticised for their often illogical sequencing, their level of overlap, the gaps that they contain, and most of all, the difficulty people experience in remembering them all. Researchers have made efforts to evolve TRIZ principles. Osborn [1993] simplified the 40 classical principles and developed the SCAMMPERR model which comprises the behaviours of "Substitution", "Combination", "Adaptation", "Magnification", "Modification", "Putting-to-another-use", "Elimination", "Re-arrangement", and "Reversal". Buzan [1993] discussed the connections between TRIZ and mind-mapping and the need to think in Time and Space. Mann [2002] combined Neuro-Linguistic Programming (NLP) thinking and the SCAMMPERR model. Nakagawa [2001] simplified TRIZ into five general elements in Unified Structured Innovative Thinking (USIT) and constructed a new problem solving process. Besides assisting concept generation, this model also applies the TRIZ philosophy to problem specification.

### 2.4.2 Conceptual design with TRIZ

Since TRIZ is a great tool for generating inventive ideas, it has been applied in the design process. Valeri Souchkov [1998] developed a TRIZ-based Conceptual Design model, which integrated inventive principles and evolutionary trends for supporting the following two phases of conceptual design: concept generation and concept evolution. Darrel Mann [2002] built a general systematic creativity process consisting of four phases: define, select tool, evaluate, and generate solutions, which can be applied in the phase of concept generation. Different TRIZ tools have been discussed to fit in each phase in his systematic creativity process. Toru Nakagawa [2002] simplified TRIZ into five general elements in USIT and constructed a new Problem Solving Process. As well as assisting concept generation, this model also applies TRIZ philosophy to problem specification. As one of the best TRIZ-derived tools, USIT combines the features of ASIT and TRIZ. It also has a significant feature which applies the idea of "closed world (object-attributes-functions)" and certain techniques of TRIZ in the problem analysis phase of conceptual design.

## 2.5 Summary

In this chapter the research relevant to the work presented in this thesis has been surveyed. An overview of the conceptual design process has been given. Special attention was paid to the design representation phase and concept generation phase. The existing techniques, used to aid each phase of conceptual design, were outlined.

TRIZ and KBS were selected to assist creative design concept generation.

This chapter also reviewed the literature relating to conflict resolution in conceptual design. It can be seen that negotiation has been a widely accepted approach for conflict resolution, while the no-compromise approach, such as TRIZ, has never been applied to resolving conflicts in design. Therefore, there is an opportunity for exploring the combination of both the negotiation and the no-compromise approaches into conflict resolution in design.

Finally, a review of TRIZ and its application in conceptual design has been given. It can be seen that the existing TRIZ-based design models have been developed or modified based on the original TRIZ inventive principles. However, the collection of inventive principles is limited and fixed. Therefore, there is a demand to restructure and evolve TRIZ CM and IP, make them flexible, and apply the modified principles to conceptual design.

# Chapter 3. Modification of TRIZ inventive principles

## 3.1 Preliminaries

Design is a process involving the application of human intelligence. The intuition and experience of the designer plays a significant role which cannot be fully replaced by any current computer-aided tools or artificial intelligence technology. Conceptual design is the first phase of the design process. Most basic functions of a new product and the majority of design solutions are generated in this critical stage, which will affect the attributes in the later detailed design phase. Conceptual design is a very important task in computer-aided design (CAD) [Wang 1994], particularly when new and innovative products are to be created. There is a need for a methodology to help designers to solve inventive problems and to generate solutions during conceptual design.

This chapter justifies the application of I-Ching to evolving TRIZ principles and discusses the representation of the inventive principles of the TRIZ theory as symbolic expressions. Each expression comprises two sets of relations between predicates and objects respectively. The expressions are arranged in a two-dimensional matrix, the Inventive Principles Matrix (IPM), according to the predicates and objects that they contain. The structure of the IPM is inspired by the I-Ching system. A conceptual design problem is translated into a set of queries stating the objectives of the design and the constraints on the solution. The queries, also represented in a symbolic form, are then used to search the IPM for suitable solutions. Lastly, the chapter describes an application of the proposed IPM and query technique to the problem of designing a bushing.

# 3.2 I-Ching

## 3.2.1 Common points between I-Ching and TRIZ

I-Ching has been selected to evolve TRIZ principles because of the common points between I-Ching and TRIZ (Table 3.1). First, the two theories both aim to improve creativity and inventiveness. Second, both of the theories are generated in the induction method. The difference here is that I-Ching offers a set of more complete and systematic laws in a general way which mirrors the universe, while that of TRIZ contains significantly greater richness in terms of its detailed applications, (40 principles), especially to the technical world. Some benefits may therefore be expected to emerge when the two approaches are combined.

|  | **TRIZ** | **I-Ching** |
|---|---|---|
| Description | A Russian system of toolkits to improve inventiveness | A Chinese philosophy system of creativity |
| The method of generating theory -Induction method | Abstracted from hundreds of patents in USSR patent database. | Concluded from kinds of natural phenomenon and social phenomenon around people. |
| Dialectics Philosophy -Contradictions | Inventive problems are solved by removing or eliminating the contradictions. | The process of problem resolution is the process of contradiction resolution. |

Table 3.1 Comparison between I-Ching and TRIZ

More importantly, these two methods are based on the same philosophy. I-Ching is the first book in the world to describe dialectical thinking, which was summarised as the three laws of dialectics by Engles almost 3000 years later. TRIZ is a successful theory for solving inventive problems by using dialectic thinking, which has been shown to be useful for creative design thinking [Quinsan 2002]. In general, I-Ching is the first theory that conforms to the three laws of dialectics [Engels 1968] underpinning TRIZ.

### 3.2.2 Philosophy of creativity

The term "dialectic" was first mentioned by the Greek philosophers Socrates and Plato, and was used as the logical method of philosophy in the Socratic dialectical method of cross-examination. Hegel [1896] reintroduced the idea of dialectics and presented a dialectically dynamic model of nature and of history as a fundamental aspect of the nature of reality. In the mid-19th century two followers of Hegel, Karl Marx and Frederick Engels, took Hegel's ideas and transformed them into a philosophical tool for analysing history, nature, and for effecting social change. Engels also formulated three laws of dialectics in his book Dialectics of Nature [Engels 1968].

Although I-Ching did not define the term "dialectic", its philosophy was full of the idea of dialectics a thousand years before the term "dialectic" emerged. The philosophy of I-Ching theory covers three laws of dialectics: The Law of the Unity of Opposites, The Law of Transformation of Quantity into Quality, and The Law of the Negation of Negation.

These laws act in all fields of objective reality and thought, but their development is different for each field. Therefore, every new developing science must follow these laws [Petrov 2002]. TRIZ also applies these laws for solving inventive problems.

### 3.2.2.1 The Law of the Unity of Opposites

The law of the unity of opposites can be explained simply as "Everything is made of opposing sides." For example, an atom is the unity of nucleus and electrons which have opposite charges. This law is the core of dialectics, which was described as the basic idea in I-Ching theory. The ancient Chinese philosophers regarded the universe as being composed of two opposite forces (Figure 3.1), Yin (negative) and Yang (positive).

Figure 3.1 Taiji diagram:

Yin (negative, black parts) and Yang (positive, white parts)

This law is also called "the law of contradiction". TRIZ happens to have the same view by focusing on the concept “contradiction” for solving inventive problems. Athuller [1996] says, “Every inventive problem is made of contradictions. The inventive problem is solved by removing or eliminating the contradictions.”. If a designer improves one system’s parameter it makes another parameter worse. These two parameters make up of a pair of contradictions, corresponding to the forces yin and yang in I-Ching theory. A whole inventive problem is the unity of two opposite parameters.

### 3.2.2.2 The Law of Transformation of Quantity into Quality

The Law of Transformation of Quantity into Quality defines a general mechanism of evolution. A simple example from the physical world might be the heating of water: a one degree increase in temperature is a quantitive change, but at 100 degrees there is a qualitative change - water to steam. As with the other two laws of dialectics, this law can be applied in the development of everything, including the evolution of a technical system, which is normally described as a four-stage S-curve diagram. Consider the example of new product development. The start point may be only an idea. Quantitive change happens at the first and second stages when the information of detailed design schemes, diagrams, or models is continuously generated. A qualitative change occurs at the third stage when a design task is finished and a physical new product produced. The second law of dialectics and the S-curve evolution were also indicated in I-Ching theory [Raymond 1971]. (Table 3.2 and Figure 3.2)

| **Name** | **Chinese symbol** | **Meaning** | **Line of life of technical system** | **Stages of General development** |
|---|---|---|---|---|
| Yuan | 元 | Origination | Mono-system | Birth |
| Heng | 亨 | Penetration | Bi-system | Growth |
| Li | 利 | Harmony | Poly-system | Mature |
| Zhen | 贞 | Firmness | Complex system | Retirement |

Table 3.2 The Four Attributes of Spirit in I-Ching

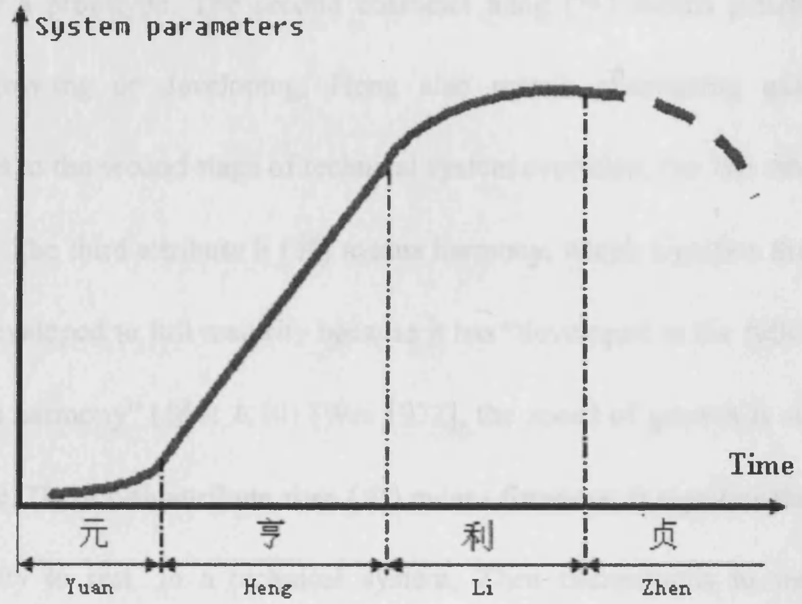


Figure 3.2 S-curve of system growth

The Qian(乾) hexagram represents the creative principle of the universe. The first sentence in King Wen's Tuanci (彖辞)， which is also the first sentence in the text part of I-Ching, is: (乾，元 亨 利 贞) Qian (yuan heng li zhen), which represents the attributes of origination, penetration, harmony, and firmness.

"yuan heng li zhen" denotes the four main stages in every cycle of system evolution. The first character yuan (元) means origin. In a technical system, yuan represents the birth of the system. The initial cause of developing a new system is a new idea, concept, or a prototype. The second character heng (亨) means penetrating and it signifies growing or developing. Heng also means penetrating quickly, which corresponds to the second stage of technical system evolution, the fast development of the system. The third attribute li (利) means harmony, which signifies that the system has been developed to full maturity because it has "developed to the fullest extent and reached the harmony" (保合太和) [Wei 1977], the speed of growth is slowing down in this stage. The fourth attribute zhen (贞) means firmness. It signifies the withdrawal from activity to rest. In a technical system, Zhen corresponds to the last stage, retirement of the system.

TRIZ presents 35 trend lines of system evolution. These trend lines indicate the direction in which a system evolves and which conform to an S-curve and the law of transformation of quantity to quality. Take the example of the trend line No. 4 "object segmentation”. The first three stages of this trend are “monolithic *solid*”, “segmentaed *solid”*, and “particulate *solid*”. The transformation from quality to quantity occurs at the fourth stage as the physical phase of the object is transformed from “*solid*” to “*fluid*”.

### 3.2.2.3 Law of the Negation of the Negation

The last law is also about evolution, but in a more panoramic perspective. The essence of this law is that a process of evolution consists of a series of relative repetitions. But each repetition takes place at a higher level of evolution. Therefore, it is also called a spiral-shaped evolution. One of the obvious examples of this law is that of fashion trend.

I-Ching reflects this law by presenting the text of “Zhen xia qi yuan”(贞下起元). As mentioned in the last section, if “Zhen” is the last stage of an S-curve, then what will happen after completion of the last attribute Zhen? The answer underlies the well-known explanatory sentence which appears in most commentaries of the I-Ching; “zhen xia qi yuan”, which may be translated as: “after zhen, yuan arises again”, as shown in Figure 3.3.

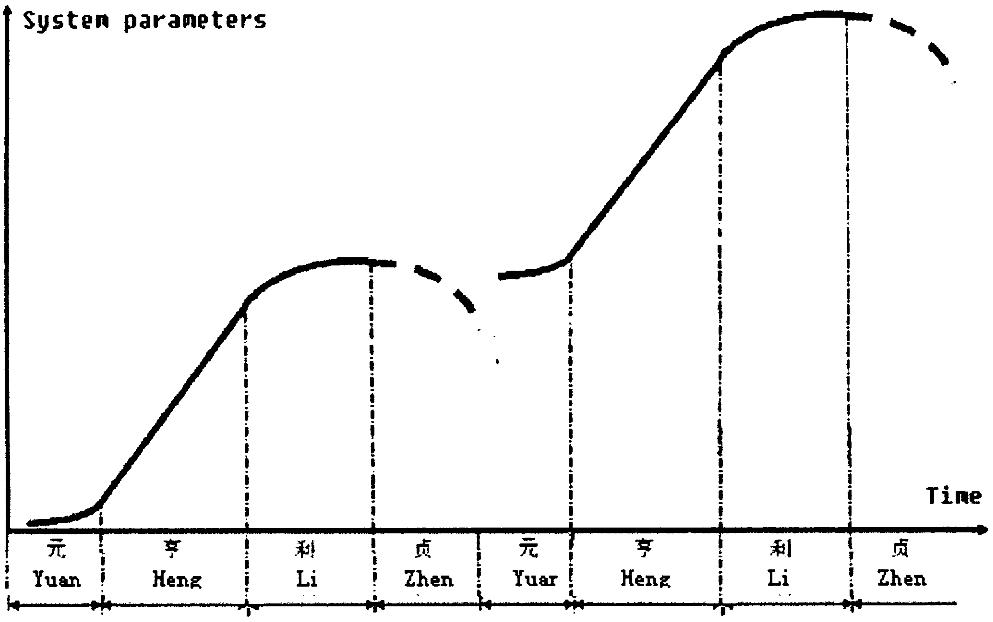


Figure 3.3 New S-curve (Negation of negation)

This unceasing trend correctly mirrors the evolution of a technical system. Once the old system has entered its retirement period, a high level functional system, a new generation system will come up, and follow the high level S-curve from the start point, yuan.

The law of the negation of the negation can be expressed as a few repetitive s-curves. Each s-curve takes place at a higher level of evolution by using new elements, materials, and technologies (Figure 3.3). For example, in TRIZ Trend line No.2 “space segmentation”, the first four stages are: “monolithic solid”, “hollow structure”, “structure with multiple hollows”, and “capillary/porous structure”. The fifth stage is “Porous strucutre with active elements”. If the first four stages follow the development of the first s-curve, then the new s-curve arises at a higher level from the fifth stage because new elements - *active elements* are added to the structure in the fourth stage.

In general, the philosophy of TRIZ is “Inventive problems can be solved by removing or eliminating the contradictions.” TRIZ researchers [Vladimir 2002; Toru 2001] have noted that the basic philosophy of TRIZ uses dialectic thinking. Dialectic thinking is not a widely used thinking method in western countries, but it has been proved that dialectic thinking does work very well for solving inventive problems [Mann 1998]. As the origin of dialectic thinking , I-Ching might be a good tool for improving TRIZ and aiding people to increase their creativity.

## 3.3 Newly structured symbolic expression of inventive principles

### 3.3.1 Formation of I-Ching-based Framework

The set of 40 Inventive Principles of classical TRIZ discovered by Altshuller is a useful creativity tool for a variety of problem solving situations [Fey 2005]. However, they are criticised frequently for their often illogical sequencing, their level of overlap, the gaps that they contain, and most of all, the difficulty people experience in remembering them all. In order to make the principles easy to remember, previous researchers tried to simplify or restructure them. Matrices are regarded as a possible way to reorganise and represent these principles systematically. Althuller's classical 39×38 TRIZ contradiction matrix is too large and can confuse the user [Altshuller 1988]. At the same time, the 3×5 matrix developed by Mann is too simple to represent the meaning of the principles comprehensively. Thus, there is a need for a new matrix of appropriate dimensions to express and evolve inventive principles.

Compared with Althuller's 39×38 matrix and Mann's 3×5 matrix, the I-Ching 8×8 hexagram matrix is suitable for representing TRIZ principles. The I-Ching hexagram matrix has a systematic structure. A Cartesian coordinate system can be added to the matrix to illustrate the sequence of placement of parameters on the first row and the first column of the 8×8 matrix.

| Basic ideas | Corresponding principles |
|---|---|
| Segment | P1a,P1b,P15b |
| Take out | P2 |
| Merge | P5,P6,P7, |
| Introduce (make use of) | P8b,P39b,P18a,P31,P4a,P24a,P14,P17,P13a, P18,P23a,P32c,P36,P38 |
| Replace-different | P39a,P30a,P29b,P28a,P28b,P38a |
| Transform from one side to its opposite | P28c,P14c,P13c,P22a |
| Modify | P4b,P12,P32a,P32b,P32d,P35 |
| Do more or less | P4c,P15d,P16,P22 |
| Re-orient | P17d |
| Replace-same | P11,P26,P27 |
| Precede | P10b,P9,P10a |
| Segment time | P19a |
| Change magnitude | P19b,P20 |
| Change frequency | P19b,P21a,P18b |
| Make the object different | P19c,P33a |
| Local quality | P3c |
| Transform from uniform to non-uniform | P3,P40 |
| Dynamics | P15a,P23b |
| Temporary | P24b,P34 |
| Self-service | P25a |
| Boundary | P30b |
| Magnify size (space) | P37 |

Table 3.3 Original basic ideas derived from 40 principles of invention

### 3.3.2 X-axis parameters

The X-axis was set for representing the fundamental ideas for inventive problem solving which are extracted from 40 inventive principles. Table 3.3 shows 23 original basic ideas derived from the 40 principles of invention.

However, as 23 simplified principles are still too complex to remember and apply, it is necessary to simplify and reorganise them again. Therefore, the I-Ching framework has been used to help this reorganisation. It has been chosen because it is a novel, original, but also a fundamental method that can reflect the process of generating critical and basic ideas for solving problems for human beings.

In Figure 3.4, the I-Ching philosophy proposes the Limitless ("Wuji") which represents the source of creativity and produces two forms, namely "yin" (negative) and "yang" (positive). The two forms produce four phenomena, known as "small yang", "great yang", "small yin", and "great yin". Then, the four phenomena act on the eight trigrams. The I-Ching trigrams comprise three lines: a line may be broken, in which case it represents "yin", or unbroken, in which case it represents "yang".

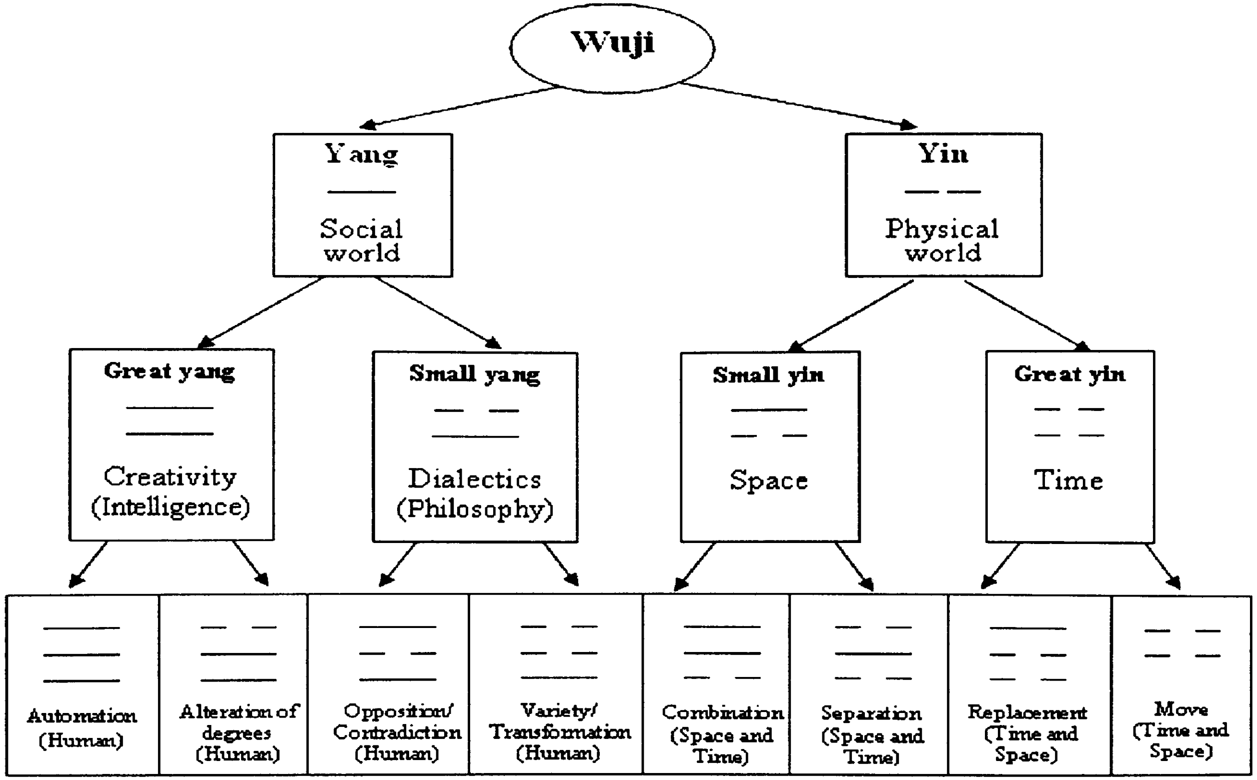

Wuji
Yang
Social world
Yin
Physical world
Great yang
Creativity (Intelligence)
Small yang
Dialectics (Philosophy)
Small yin
Space
Great yin
Time
Automation (Human)
Alteration of degrees (Human)
Opposition/ Contradiction (Human)
Variety/ Transformation (Human)
Combination (Space and Time)
Separation (Space and Time)
Replacement (Time and Space)
Move (Time and Space)

Figure 3.4 Assignment of parameters on x-axis

Representing eight fundamental problem-solving behaviours, the eight trigrams are derived from Wuji in three stages (Figure 3.4). The first four behaviours are "automation", "alteration of degrees", "transformation", and "contradiction". These involve uncertainty and human decisions. The last four elements of "combination", "segmentation", "replacement", and "move" are relatively explicit behaviours. The three lines in an I-Ching trigram correspond to three aspects of each basic solution. These can be time, space, and other human factors such as personal belief, value, and perception. The eight behaviours can thus be divided further into twenty-four sub-parameters. The x-axis parameters and their meaning are shown in Table 3.4.

| General idea | | Sub-idea | |
|---|---|---|---|
| | | Key words | Meaning |
| 1 | Automation | a. Dynamics | a. Make the system dynamical |
| | | b. Adjustability | b. Make the system adjustable |
| | | c. Optimisation | c. Make the system optimised |
| 2 | Alteration of Degrees | a. Increase | a. Increase the degree |
| | | b. Decrease | b. Decrease the degree |
| | | c. Limit | c. Make use of full load |
| 3 | Contradiction | a. Inversion-positive | a. Transform from bad to good |
| | | b. Inversion-negative | b. Transform from good to bad |
| | | c. Recurrence | c. Switch the system from one state to the opposite state and back. |
| 4 | Transformation | a. Uniformity | a. Make the system homogeneous |
| | | b. Non-uniformity | b. Make the system non-homogeneous |
| | | c. Balance | c. Make the system balance |
| 5 | Combination | a. Merging (T) | a. Merge (in time) |
| | | b. Merging (S) | b. Merge (in space) |
| | | c. Addition | c. Add new elements (in space) |
| 6 | Separation | a. Partition (T) | a. Segment (in time) |
| | | b. Partition (S) | b. Segment (in space) |
| | | c. Removal | c. Take off elements (in space) |
| 7 | Replacement | a. Substitution | a. Replace with different objects (in space) |
| | | b. Reservation | b. Backup the object for future use (in time) |
| | | c. Restoration | c. Re-introduce the element after it has been out of use (in time) |
| 8 | Movement | a. Move | a. Move (in space) |
| | | b. Precede | b. Precede (in time) |
| | | c. Delay | c. Delay (in time) |

Table 3.4 Parameters and sub-parameters in x-axis

### 3.3.3 Y-axis parameters

Normally, a solution to a problem is composed of two parts from a grammar point of view: the predicate and the object. The x-axis parameters mentioned above relate to behaviours and play the role of the "predicate" in a sentence. The 8 y-axis parameters play the role of the "object" in a sentence.

Y-axis parameters are extracted not only from 40 TRIZ principles in terms of the object of an action, but also obtained by referring 35 evolution trends and 76 standards in TRIZ theory. They can represent basically most of the objects involved in solutions for solving inventive problems.

The order of y-parameters is from general to specific in terms of an abstract level, as shown in Figure 3.5.

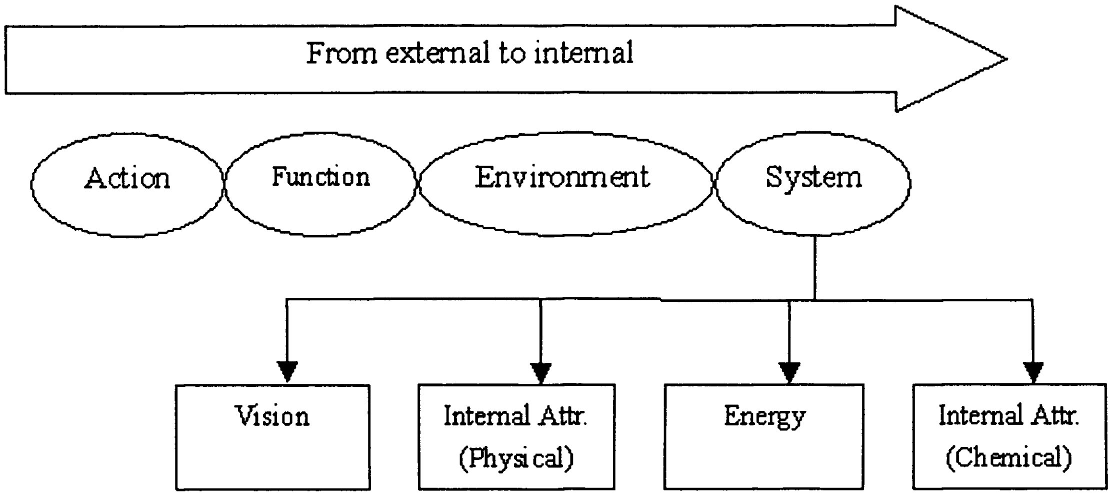


Figure 3.5 Assignment of parameters on y-axis

The 8 parameters on the y-axis can also be divided into two groups. The first four parameters are "action", "function", "environment", and "system". These are general terms with a high abstraction level. They are arranged along the y-axis according to whether they represent external or internal aspects of attributes. The last four parameters are re-organised from 12 types of fields in TRIZ [Mann 2002]. A "field" is defined as any source of energy within a system. These four parameters are arranged from macro to micro aspects into: "measurement", "physics", "energy", and "micro level". As with the x-axis, each one of the main parameters on the y-axis is divided into 3 sub-parameters. However, the y-axis parameters are not a two-level hierarchical structure. Each one of the 24 sub-parameters can be divided again to express a more specific attribute when needed. The details of the sub-parameters are shown in Table 3.5.

|  |  |  |  |  |
|---|---|---|---|---|
| 1. Action | 1a. Useful action |  |  |  |
|  | 1b. Harmful action |  |  |  |
|  | 1c. Anti-action |  |  |  |
| 2. Function | 2a. Positive function | 2aa. Low value/cost |  |  |
|  | 2b.Negative function | 2ba. High value/cost |  |  |
|  |  | 2bb. Short living |  |  |
|  | 2c. Redundant function |  |  |  |
| 3. Environment | 3a. Resource |  |  |  |
|  | 3b. Waste |  |  |  |
|  | 3c. Interface | 3ca. 'Intermediary' |  |  |
|  |  | 3cb. Feedback |  |  |
|  |  | 3cc. Gaps |  |  |
| 4. System | 4a. Present system |  |  |  |
|  | 4b. Sub-system |  |  |  |
|  | 4c. Super system |  |  |  |
| 5.Measurement | 5a.Vision | 5aa. Dimension | 5aaa. Size | 5aaaa. Direction |
|  |  |  |  | 5aaab. Side/height |
|  |  |  |  | 5aaac. Area |
|  |  |  | 5aab. Shape | 5aaba. Asymmetry |
|  |  |  |  | 5aabb. Curve |
|  |  |  |  | 5aabc. Non-linear |
|  |  | 5ab. Structure | 5aba. Porous/hole |  |
|  |  |  | 5abb. Shell/film |  |
|  |  | 5ac. Colour | 5aca. Ultraviolet/infrared |  |
|  |  |  | 5acb. Transparent |  |
|  | 5b. Other senses | 5ba. Acoustic | 5baa. Resonance |  |
|  |  | 5bb. Taste |  |  |
|  |  | 5bc. Smell |  |  |
|  |  | 5bd. Touch |  |  |
|  | 5c. Mechanical means |  |  |  |
| 6.Physics (macro level) | 6a. Dynamics | 6aa. Motion | 6aaa. Rotation |  |
|  |  |  | 6aab. Vibration | 6aaba. Frequency |
|  |  |  |  | 6aabb. Magnitude |
|  |  |  |  | 6aabc. Damping |
|  |  | 6ab. Force | 6aba. Pressure |  |
|  |  |  | 6abb. Lift/buoyancy |  |
|  |  |  | 6abc. Centrifugal forces |  |
|  |  | 6ac. Time |  |  |
|  | 6b. Materials | 6ba. Density |  |  |
|  | 6c. Physical phase | 6ca. Solid |  |  |
|  |  | 6cb. Liquid |  |  |
|  |  | 6cc. Gas |  |  |
| 7.Energy | 7a. Thermal | 7aa. Temperature |  |  |
|  |  | 7ab. Thermal expansion/contraction |  |  |
|  |  | 7ac.Radiant | 7aca. Emissivity |  |
|  | 7b. Electronics |  |  |  |
|  | 7c. Magnetic |  |  |  |
| 8. Micro level | 8a. Chemistry | 8aa. Concentration |  |  |
|  |  | 8ab. Atmosphere | 8aba. Oxygen |  |
|  |  |  | 8abb. Ozone |  |
|  |  |  | 8abc. Inert/neutral atmosphere |  |
|  |  | 8ac. Ionisation |  |  |
|  | 8b. Biology |  |  |  |
|  | 8c. Nuclear(physics) |  |  |  |

Table 3.5 Y-axis parameters

### 3.3.4 Representation of principles

As mentioned before, each inventive principle is a method for solving problems. This normally consists of a behaviour and a system or an attribute, expressed separately by parameters on the x-axis and y-axis. In some cases, the solution is complex and has to be represented by combining two or more parameters and their relationship.

Therefore, each single concept $C_m$ can be described basically as an expression of two sets $R_x$ and $R_y$ of predicates and objects, as shown in Eq 3.1. This equation can be spread as Eq 3.2.

$$C_m=R_x[a_{x1},a_{x2}...a_{xi}]R_y[b_{y1},b_{y2}...b_{yj}](i, j=1, 2...) \tag{3.1}$$

$$C_m=[a_{x1}\ R_{x1}\ a_{x2}\ R_{x2}\ ...R_{x(i-1)}\ a_{xi}]\ [b_{y1}\ R_{y1}\ b_{y2}\ R_{y2}\ ...R_{y(j-1)}\ b_{yj}](i, j=1, 2...) \tag{3.2}$$

In Eq 3.1, $R_x$ and $R_y$ separately show the relationships between parameters on the x-axis and the y-axis. $R_x$ and $R_y$ can be one of the relationships $R_1$, $R_2$, $R_3$, or $R_4$ in Table 3.6.

| **R** | **Symbol** | **Meaning** | **Example** |
|---|---|---|---|
| $R_1$ | $\vee$ | Or | $a_x \vee a_y$ |
| $R_2$ | $<$ | Part of /in | $a_x < a_y$ |
| $R_3$ | $>$ | Comprise | $a_y > a_x$ |
| $R_4$ | $()$ | Sequence | $[a_x](C_m)$ |
| $R_5$ | $\rightarrow$ | In order to | $C_m \rightarrow C_n$ |
| $R_6$ | $\leftarrow$ | According to | $C_m \leftarrow C_n$ |

Table 3.6 Six types of relationship

As shown in Table 3.6, the first type is the logic relationship "OR" represented by "□". $R_2$ and $R_3$ are "inclusion" relationships, which can have three meanings under different conditions: 1) $a_x$ is part of $a_y$; 2) $a_x$ is an attribute of $a_y$; 3) $a_x$ is the value of an attribute of $a_y$. $R_1$, $R_2$, and $R_3$ are used to represent a concept. The other two relationships in Table 1 are employed in expressions of inventive principles. $R_4$ represents the sequence relationship: concept $C_m$ is implemented first, then $a_x$ acts on $C_m$. $R_5$ is an objective relationship: concept $C_m$ is performed to satisfy goal $C_n$. $R_6$ is the opposite of $R_5$, where $C_n$ is the basis or the reason of $C_m$. Eqs 3.3, 3.4 3.5 and 3.6 show the three possible expressions of an inventive principle.

$$P_1=C_m \qquad (m=1, 2...) \tag{3.3}$$

$$P_2=[a_x](C_m) \qquad (m=1, 2...) \tag{3.4}$$

$$P_3= C_m \rightarrow C_n \qquad (m, n=1, 2...) \tag{3.5}$$

$$P_4= C_m \leftarrow C_n \qquad (m, n=1, 2...) \tag{3.6}$$

Eqs 3.1 to 3.6make up the proposed symbolic expression system for inventive principles. These principles are mapped into corresponding cells in a matrix as shown in Figure 3.6. All the classical 40 inventive principles, including sub-principles with details, can be represented in twelve forms using the symbolic expression system (Table 3.7). These twelve forms of symbolic expressions can be transformed into natual language according to certain rules. The details of the translation rules will be described in chpater 5.

|  | 1.Automation<br>1a.dynamics<br>1b.adjustability<br>1c.optimisation | 2.Degree<br>2a.increase<br>2b.decrease<br>2c.limit | 3.Contradiction<br>3a.opposition<br>3b.inversion<br>3c.recurrence | 4.Transformation<br>4a.uniformity<br>4b.non-uniformity<br>4c.difference | 5.Combination<br>5a.merging (T)<br>5b.merging (S)<br>5c.addition | 6.Separation<br>6a.partition (T)<br>6b.partition (S)<br>6c.removal | 7.Replacement<br>7a.substitution<br>7b.reservation<br>7c.restoration | 8.Move<br>8a.move<br>8b.precede<br>8c.delay |
|---|---|---|---|---|---|---|---|---|
| 1.Action<br>1a.useful action<br>1b.harmful action<br>1c.anti-action |  | [x2a∨x2b][y1]<br>[x2b][y1b] | [x3][y1]<br>[x1c][y3>y2b]<br>→[x3b][y2a] |  | [x5][y1∨y2∨y4]<br>[x5c>x8b][y1a]<br>[x5c][y1b] | [x6a][y1]<br>[x6a>x2c][y1b] |  | [x8b][y1c] |
| 2.Function<br>2a.positive function<br>2b.negative function<br>2c.redundant function | [x1a>x4b][y2<y4b] | [x2a][x2b<y4] |  | [x4b][y2<y4b]<br>[x4b][y2<y4] | [x5c][y2a<y5aba] |  |  |  |
| 3.Environment<br>3a.resource<br>3b.waste<br>3c.interface | [x4b][y1a<y8cc]<br>[x1b][y3cb]<br>[x1c][y3b]<br>[x1c][y3a] |  |  | [x4b][y3]<br>[x4c>x4a][y3] | [x5c][y3cb]<br>[x5c][y3ca<y4] | [x6c>x8c][y3ca] |  |  |
| 4.System<br>4a.present-system<br>4b.sub-system<br>4c.super-system | [x1b>x6b][y4]<br>[x1c][y4>y5aabb<br>]<br>[x1a][y4]<br>[x1b][y4]<br>[x1][y4]<br>[x1c][y4c] | [x2a]([x6][y4])<br>[x5b]([x2a][y4>y5aba])<br>[x2b]([x4][y7]) |  | [x4b][y4]<br>[x4a][y4b] | [x5b][y4>y5aba]<br>[x5b][y4>y6abb]<br>[x5a>x2c][y4]<br>[x5c]([x7b][y4]) | [x6b][y4]<br>[x6c][y4b>y2b]<br>[x6b>x1b][y4]<br>[x6c>x1][y4b] | [x7b][y4>y2c]<br>[x7a][y4>y2c]<br>[x7c][y4b]<br>[x7b][y4>y5] | [x8b>x1b][y4] |
| 5.Measurement<br>5a.optial<br>5b.another sense<br>5c.mechanical means | [x1c][y5aaab<4]<br>[x1c][y5aca]<br>[x1b][y5abb] | [x2a][y5aaba]<br>[x2a][y5aaa]<br>[x2a][y5aaab] | [x3a][y5aaa<y4] | [x4c][y5aa]<br>→[x1c][y5aaba<y3]<br>[x4c][y5ab]<br>[x4c][y5ac<y4]<br>[x4c][y5ac<y3]<br>[x4c][y5acb<y4]<br>[x4c][y5acb<y3]<br>[x4c][y7aca<y4] | [x5c][y5aaba]<br>[x5c][y5aabb]<br>[x5c][y4>y5aba]<br>[x5c][y4>y5ac] |  | [x7b][y4>y5a]<br>[x7a][y5a∨y5b] | [x8a][y5aab<y4] |
| 6.Physics<br>6a.dynamics<br>6b.materials<br>6c.physical phases | [x1c][y6abb<y6cb]<br>[x1c][y6aba>y6aa]<br>[x1c][y6cb∨y6cc]<br>[x1c]((x4c)[y6c]) | [x2a][y6aa]<br>[x2b][y6aabc]<br>[x2a]([x6a][y5])<br>[x2c]([x6a][y5]) | [x3c][y6aa]<br>[x3c][y6aaa] | [x4c][y6aaba ∨ y8aabb]<br>→[x1c][y3]<br>[x4c][y6c]<br>[x4c][y6aba] | [x5c>x8c][y6aba<y4]<br>[x5c∨x1c][y6abc]<br>[x5c][y6aab]<br>[x5][y6aab] | [x6a][y6aab<y4]<br>[x6c][y6aabc<y4] |  |  |
| 7.Energy<br>7a.thermal<br>7b.electronics<br>7c.mechanics | [x1c][y7ab] |  |  | [x4c][y7aa]<br>[x4][y7ab]<br>[x4b][y6b] → [x1a][y4] | [x5c][y7b∨y7c]<br>[x5b][y6b<y4] |  | [x7a][y8aba]<br>[x7a][y8abc] |  |
| 8.Micro level<br>8a.chemistry<br>8b.biology<br>8c.nuclear | [x1c]([x2c][y8aba])<br>[x1c][y8ac]<br>[x1c][y8aba>y8ac]<br>[x1c][y8abb]<br>[x1c][y8] |  |  | [x4c][y8aa] | [x5c][y4b>y8abc] |  |  |  |

$a_x$

$b_y$

Figure 3.6 Inventive Principles Matrix (IPM)
(New principles are shown in grey background)

| **Equation** | **Expression** | **Example** |
|---|---|---|
| $P_1=C_m$ | $[a_x][b_y]$ | P1a: [x6b][y4] |
| | $[a_{x1}\ R_x\ a_{x2}][b_y]$ | P15b: [x6b>x1a][y4] |
| | $[a_x]\ [b_{y1}\ R_y\ b_{y2}]$ | P17d: [x1c][y6aaa<y4] |
| | $[a_{x1}\ R_x\ a_{x2}][b_{y1}\ R_y\ b_{y2}]$ | P14d: [x5c□x1c][y6ab>y5aabb] |
| | $[a_x]\ [b_{y1}\ R_{y1}\ b_{y2}\ R_{y2}\ b_{y3}]$ | P5a: [x5][y1□y2□y4] |
| | $[a_x]\ [b_{y1}\ R_{y1}(b_{y2}\ R_{y2}\ b_{y3})]$ | P22a: [x3a][y1b□(y4>y2b)] |
| | $[a_{x2}\ R_{y1}(a_{x2}\ R_{y2}\ a_{x3})]\ [b_y]$ | P1b: [x1b>(x5b□x6b)][y4] |
| $P_2=[a_x](C_m)$ | $[a_x]([a_{x1}][b_y])$ | P1c: [x2a]([x6][y4]) |
| | $[a_x](\ [a_{x1}\ R_x\ a_{x2}][b_y])$ | P34a: [x8]([x1>x6c][y4b]) |
| | $[a_x]([a_{x1}]\ [b_{y1}\ R_y\ b_{y2}])$ | P17e: [x1c]([x2a][y5aaab<y4]) |
| $P_3= C_m \rightarrow C_n$ | $C_m \rightarrow C_n$ | P19b: [x4c][y6aaba□y6aabb]<br>→ [x1c][y3] |
| $P_4= C_m \leftarrow C_n$ | $C_m \leftarrow C_n$ | P4b: [x4c][y5aa] ← [y5aaba<y3] |

Table 3.7 Twelve forms of expressions

The Inventive Principles Matrix (IPM) expresses innovative principles in a format which can reduce the amount of repeated information found in the classical collection of inventive principles. It can be seen from Figure 3.6 that only ninety-three solutions are contained in the matrix.  Five solutions with redundant information have been eliminated, including Principle 35F (change other parameters) which is too vague.

## 3.4 Generation of new inventive principles

Although the matrix looks like a closed system, due to the last four parameters on the y-axis being structured in a multi-level hierarchy (see the Table 3-4), systems relating to inventive solutions can be added to this matrix in the future. Moreover, the matrix can suggest solutions to general problems. In order to solve more specific problems, the knowledge/effects database offered by TRIZ theory could be integrated by constructing an interface between it and the y-axis parameters.

Besides TRIZ inventive principles, other TRIZ and TRIZ-derived tools also contain novel solutions to problems. However, those tools are not directly expressed as solutions. The main meaning of the tools has been extracted and mapped to the matrix. The creative thinking tools adopted include the trends of evolution in TRIZ theory, and Unified Structured Inventive Thinking (USIT) which is derived from TRIZ.

### 3.4.1 Trends of evolution

There are 35 trend lines in TRIZ. The trend lines indicate the direction in which a system evolves. The technology trends were not only acknowledged as a strategic system evolution prediction tool, but also as a tool to help solve problems. [Mann 2002]

As the trend lines are useful tools for solving problems, the solutions implicated in them can be extracted and summarised. Normally each trend line can be transformed to at least one solution. Table 3.8 shows these solutions both in texts and in symbolic expressions, and the corresponding oringinal trend lines. Trend lines 28, 29, 30, and 35 are neglected because they are business (non-technical) related trends.

| Trend line | Solutions | Symbolic expressions |
|---|---|---|
| 1.Smart materials | Make passive material adaptive | [x1b][y6b] |
| | Increase degree of adaptation | [x2a]([x1b][y6b]) |
| 2.Space segmentation | Introduce hollow structure | [x5c][y5aba] |
| | Increase the number of hollows (porous) | [x2a][y5aba] |
| 3.Surface segmentation | Introduce curve | [x5c][5aabb] |
| 4.Obect segmentation | Segment an object | [x6b][y4] |
| | Introduce fluid, gas | [x5c][6cb□6cc] |
| 5,6.Macro to nano scale | Decrease size | [x2b][y5aaa] |
| 7.Webs and fibers | Increase dimension | [x2a][y5aa] |
| 8.Decreasing density | Decrease density | [x2b][y6ba] |
| 9.Increasing asymmetry | Increase asymmetry | [x2a][y5aaba] |
| 10,11.Boundary breakdown | Decrease boundary | [x2b][y3c] |
| 12,13.Geometric evolution | Increase dimension | [x2a][y5aa] |
| 14.Dynamisation | Make the system flexible | [x1b][y4] |
| 15,16.Action co-ordination | Combine actions | [x5a][y1] |
| 17.Rhythm co-ordination | Make continuous action periodic | [x6a][y1] |
| 18.Non-linearities | Increase non- linearities | [x2a][y5aabc] |
| 19-23.Mono-bi-poly | Combine systems | [x5a][y4] |
| 24.Reduced damping | Reduce damping | [x2b][y6aabc] |
| | Use light damping or un-damped control system | [x6c][y6aabc<y4] |
| 25.Increasing use of senses | Increase use of senses | [x2a][x1c][y5]) |
| 26.Increasing use of color | Increase use of color | [x2a]([x1c][y5ac]) |
| 27.Increasing transparency | Increase use of transparency | [x2a]([x1c][y5acb]) |
| 28.Customer purchase focus | Non-technical…. | |
| 29.Market evolution | Non-technical… | |
| 30.Design point | Non-technical.. | |
| 31.Degree of freedom | Increase degree of freedom | [x2a]([x1b][y4]) |
| 32.Trimming | Remove sub-systems | [x6c][y4b] |
| 33.Controllability | Introduce intermediary and feedback | [x5c][3ca□3cb] |
| 34.Reducing human involvement | Increase automation | [x2a]([x1][y4]) |
| 35.Design methodology | Non-technical | |
| 36.Reducing number of energy conversions | Make use of free and readily available sources | [x1c][y3a] |
| | Reduce number of energy conversions in energy flows within a system | [x2b]([x4][y7]) |

Table 3.8 Trend lines and corresponding solutions

Compared with TRIZ inventive principles, the solutions provided by the trend lines can guide people to think along a direction instead of considering the problem at a certain point. Moreover, some of these solutions are working in a more specific level than principles, which will help people to connect their own specific problems with solutions more effectively. For example, Principle No.15c says "if an object or system is rigid or inflexible, make it movable or adaptable". While the solution extracted from trend line 1 indicates "make passive material adaptive and increase the degree of adaptation". The object of the former solution is a general system where the object of the second solution is the material of an object. This could steer people to associate with their own problem and find the most suitable solution effectively.

### 3.4.2 USIT

TRIZ provides comprehensive principles and rules for generating solutions to inventive problems, while USIT presents a clearly defined simple procedure for solving real problems in industry. The procedure is composed of three stages, i.e. Problem Definition, Problem Analysis, and Solution Generation. The "Concept Generation Stage" is the core stage in USIT. Comprehensive principles and rules including "40 Inventive principles", "25 Trend lines", and "76 Standards of Inventive Solutions" in TRIZ have been reclassified into the framework of USIT. USIT has only five Solution Generation Methods: Object Pluralisation, Attribute Dimensionality, Function Distribution, Solution Combination, and Solution Generalisation Methods and 32 detailed sub-methods [Toru 2002].

Because these methods are used for generating solutions and emerged originally from TRIZ, they are used as a complementary source to make up and expand TRIZ inventive principles. The five main USIT solution generation methods and submethods and their symbolic expressions are listed in Table 3.9. Some of these methods are not transformed because they are too abstract.

| Main | Sub-Mehods | Expressions |
|---|---|---|
| (1)Object Pluralisation Method | (1a) Eliminate the Object (into 0). (Simplification, Trimming) | [x2b][y4] |
| | (1b) Multiply the Object (into 2, 3, ..., inf.). | [x2a][y4] |
| | (1c) Divide the Object (into 1/2, 1/3, ..., 1/inf.). | [x6b][y4] |
| | (1d) Unify multiple Objects into one. | [x5b][y4] |
| | (1e) Introduce a new/modified Object. | [x5c][y4] |
| | (1f) Introduce an Object from the Environment. | [x5c][y4<y3] |
| | (1g) Replace a solid Object with apowder, fluid, liquid, or gaseous Object. | [x7a][y4>(6cb□6cc)] |
| (2) Attribute Dimensionality Method | (2a) Deactivate/make irrelevant the harmful Attribute. | [x6c][y4>y2b] |
| | (2b) Activate/involve a new useful Attribute. | [x5c][y4>y2a] |
| | (2c) Enhance the useful Attribute or suppress the harmful Attribute. | [x2a][y2a]<br>[x2b][y2b] |
| | (2d) Introduce/enhance a spatial Attribute or distribute/vary in space a harmful or useful Attribute or Attribute's value. | [x2a□x5c][y5aa] |
| | (2e) Introduce/enhance a temporal Attribute or distribute or vary in time a harmful or useful Attribute or Attribute's value. | [x2a□x5c][y6ac] |
| | (2f) Change the phase, utilise the phase change, or change the inner-structure of the Object. | [x4][y6c] |
| | (2g) Utilise Attributes/properties at the micro level. | [1c][y4>y8] |
| | (2h) Improve the properties/performance of the system as a | [1c][y4a] |
| (3)Function Distribution Method | (3a) Reassign the Function to a different Object. | [x8a][y2] |
| | (3b) Divide the compound/multiple Functions and assign them to different Objects or different parts of an Object. | [x6b][y2] |
| | (3c) Unify multiple Functions and assign the unified Function to an Object. | [x5b][y2] |
| | (3d) Introduce a new Function and assign it to an Object. | [x5c][y2] |
| | (3e) Distribute/vary the Function in space or utilise the spatial distribution or motion or vibration Function. | [x4b][y2>y5aa]<br>[x1c][6aa□y6aa] |
| | (3f) Distribute/vary the Function in time. | [x4b][y2>y6ac] |
| | (3g) Realize the detection/measurement Function. | [x2a]([x6a][y5] |
| | (3h) Introduce/enhance the adapting or coordination or control Function. | [x5c]([x1b][y4]) |
| | (3i)Achieve the Function with a different physical principle. | Too abstract |
| (4)Solution Combination Method | (4a) Combine solutions functionally. | [x5b][y1a>y2] |
| | (4b) Combine solutions spatially. | [x5b][y1a] |
| | (4c) Combine solutions temporally. | [x5a][y1a] |
| | (4d) Combine solutions structurally. | [x5b][y1a>5ab] |
| | (4e) Combine solutions at the principle level. | Too abstract |
| | (4f) Combine solutions at the super-system level. | [x1c][y4c] |
| (5)Solution Generalisation Method | (5a) Generalise/specify the solution for associative thinking. | Too abstract |
| | (5b) Construct a hierarchical system of solutions. | [x1c][y5ab>(y4a□y4c)] |

Table 3.9 USIT methods and symbolic expressions

### 3.4.3 New principles with symbolic representation

Conceptual solutions are extracted from TRIZ trends of evolution and USIT and expressed in symbolic form in the previous sections. Nine expressions are different from the symbolic expressions of classical inventive principles. They are identified in grey in Figure 3.6. These new principles with examples and their symbolic expressions are shown in Table 3.10.

| References | Expression | Description | Examples |
|---|---|---|---|
| Trend line No. 24 in TRIZ | [x6c][y6aabc<y4] | Use light damping or un-damped control system | ● Aircraft flight control architecture<br>● Hydraulic systems |
| | [x2b][y6aabc] | Reduce damping | |
| Trend line No. 36 in TRIZ | [x1c][y3a] | Make use of free and readily available sources | ● Car with internal combustion engine converts chemical to heat to mechanical<br>● Locomotives |
| | [x2b]([x4][y7]) | Reduce number of energy conversions in energy flows within a system | |
| Method 2(g) in USIT | [x1c][y8] | Make use of property of micro level | ● Nanometre technology<br>● Micro-robots |
| Method 3(g) in USIT | [x2a]([x6a][y5]) | Conduct detection or measurement function as quickly as possible | ● User-adjustable lenses eliminate need for measurements by optician<br>● X-ray inspection of welds |
| | [x2c]([x6a][y5]) | Make the detection or measurement function unnecessary to skip | |
| | [x7b][y4>y5] | Introduce a measurement or detection on a copy of the object | |
| Method 4(f) in USIT | [x1c][y4c] | Solve problem in current system by combining with neighbours system and improving super-system | ● Introduce a chisel between a hammer and rock to improve the rock-breaking capability |

Table 3.10 New principles with examples

## 3.5 Case study

Compared with text, the IPM offers a more precise and systematic method for information retrieval because it stores the keywords of a solution without any redundant information. A Boolean search was selected as the search strategy; this technique retrieves those expressions which are true for a query expressed in terms of keywords and logical connections. The query will retrieve any principles of which the symbolic expression comprises ($a_{x1}$ OR $a_{x2}$ ...OR...$a_{xi}$) in the predicate expressions and ($b_{y1}$ OR $b_{y2}$...OR...$b_{yj}$) in the object expressions. Any sub-level parameters should be considered if $a_{xi}$ or $b_{yj}$ appear in the query expressions.

If no expression matches the query, then $b_y$ is modified to its super-level parameters until the result is found. One exception is that "system" $y_4$ is defined as the super-level of $y_5$, $y_6$, $y_7$, and $y_8$ because these represent specific aspects of the "system" ($y_4$).

A bushing is usually designed as an assembly containing coaxial inner and outer sleeves made from metal or another rigid material. A coaxial rubber layer can be assembled between the outer and inner sleeves. A typical design of a rubber bushing is shown in Figure 3.7 (a).

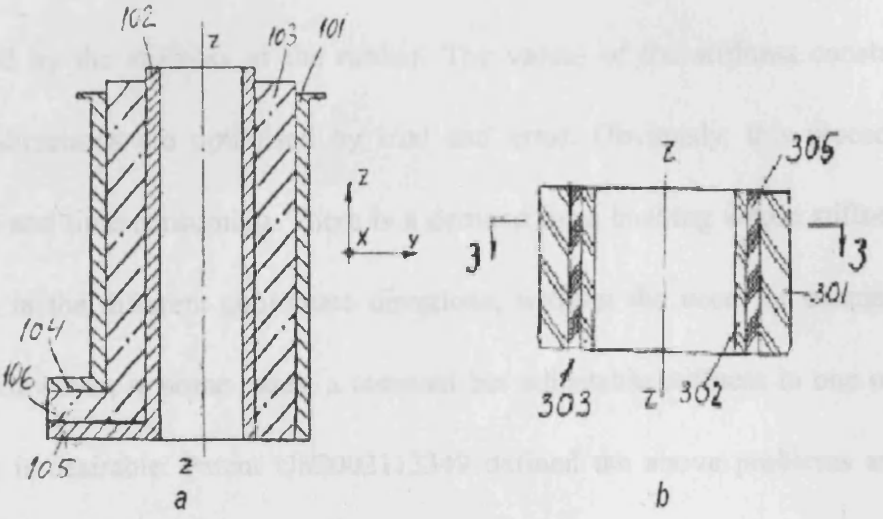


Figure 3.7 (a) Present bushing (b) Proposed bushing

The rubber insert generally provides vibration damping and various degrees of mobility between the connected mechanical components in one or more translational and/or rotational directions. The performance characteristics of the device are usually determined by the stiffness of the rubber. The values of the stiffness constants in different directions are optimised by trial and error. Obviously, this procedure is expensive and time-consuming. There is a demand for a bushing whose stiffness can be varied in the different coordinate directions, without the need for changing the bushing. However, in some cases, a constant but adjustable stiffness in one or more directions is desirable. Patent US2002113349 defined the above problems and was granted for providing solutions to them.

The query expressions for this design are shown below:

Q1= (x1a) AND (y6b OR y3ca)

Q2= (x1b) AND (y6b OR y3ca)

Query Q1 means that the required solution is a principle enabling the system to change under different conditions ((x-axis)1a. dynamics) and the system involved in this solution is rubber in a bushing ((y-axis)3ca. intermediary; (y-axis)6b. material). A similar explanation applies to Q2; the solution should enable a bushing to be adjustable ((x-axis)1b). The retrieved results are shown in Table 3.11.

| Results of Q1 | | |
|---|---|---|
| 1 | [x4b][y6b] → [x1a][y4] | √ |
| **Results of Q2** | | |
| Not found | | |
| **Results of modified Q2** | | |
| 1. | [x1b][y4] | × |
| 2 | [x8b>x1b][y4] | √ |
| 3 | [x1b>x6b][y4] | √ |

Table 3.11 Alternative solutions

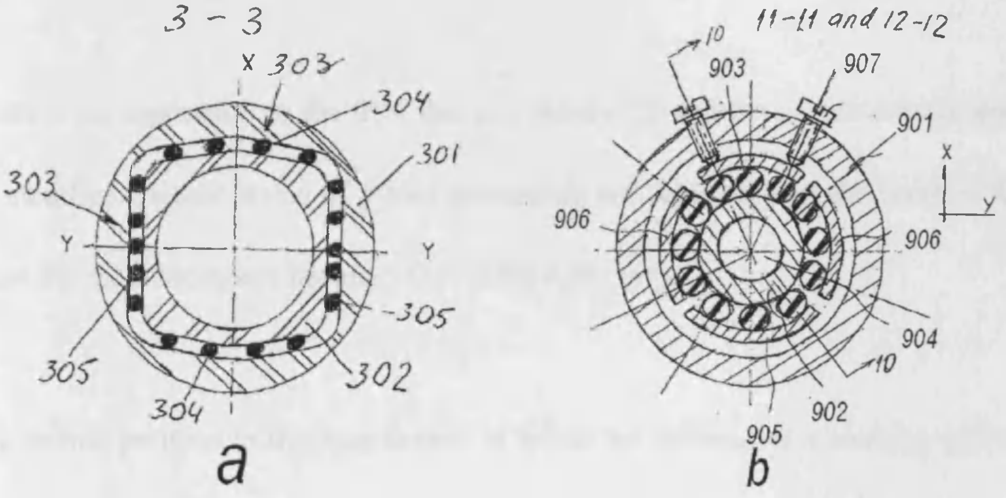


Figure 3.8(a) Proposed bushing 1 (b) Proposed bushing 2

[x4b][y6b] → [x1a][y4] is an expression of Principle 40. It can be interpreted as "to change from uniform to composite (multiple) materials" , which clearly suggests changing the existing bushing to a bushing made of multiple materials (rubber). In this case, a bushing in a variety of rubbers with different stiffnesses might be useful. Due to the different stiffnesses required in different coordinate directions, a variety of rubber cylinders with different stiffnesses can be arranged in the different coordinate directions. Figure 3.7 (b) shows an axial cross section of the proposed bushing. The outer sleeve 301 and inner sleeve 302 are separated by rubber inserts 303 comprising a plurality of rubber cylinders 304 in the x direction and 305 in the y direction whose cross section is seen as Figure 3.8 (a).

There is no expression in the IPM that can satisfy Q2 and the search criteria need to be modified. Higher levels of y-axis parameters are selected, and the result is found when the modified query becomes Q2=(x1b) AND (y4).

The second problem in this case is how to adjust the stiffness of a bushing. [x1b][y4] is an expression in IPM. Its corresponding textual meaning is "to make the system adjustable". This principle is the same as the description of requirements, but as a solution it is too general to use.

Solution [x8b>x1b][y4] suggests making the system adjustable as a pre-emptive measure. Thus, it might be possible to adjust the stiffness by changing the pre-compression on the rubber cylinders before any external radial forces are applied to the bushing. In Figure 3.8 (b), this objective is achieved by adjusting the preload on the rubber in advance with bolts (907) and holding plates (905, 906). These holding plates can be pushed forward or retracted in the radial direction, thus changing the pre-compression of the respective group of rubber elements and the stiffness in this direction. [x1b>x6b][y4] suggests making a subsystem easy to remove. This also matches the design of this patented bushing: if the rubber cylinder is no longer required to be compressed, the holding plates can be removed easily by undoing the bolts.

## 3.6 Summary

This chapter has proposed a new matrix of inventive principles that is based on I-Ching and TRIZ. The inventive principles were extended by integrating other TRIZ tools and TRIZ-derived tools. These principles are also restructured by the inspiration of I-Ching. The new symbolic expression of inventive principles enables the generation of new and innovative solutions based on TRIZ. An example of how to search the matrix and retrieve working solutions for the conceptual design of a new type of bushing was presented. The first hypothesis presented in chapter 1 has been validated as nine new inventive principles have been generated based on I-Ching-inspired symbolic expressions.

# Chapter 4. TRIZ-based design concept generation

## 4.1 Preliminaries

Conceptual design plays a very important role in the early stages of design. Conceptual design involves generating ideas or solutions, evaluating the outputs generated, and exploring the best alternatives. The generation of solutions is where engineers require creativity and imagination to achieve the design objectives while satisfying the constraints [Hyman 1998]. This step is one of the most critical in conceptual design. Many tools and techniques developed by researchers to date have focused on solution generation, including computer-aided techniques and human-oriented creative thinking tools. This chapter presents a new TRIZ-based methodology that integrates both computer-aided and human-oriented approaches to help designers solve invention problems and generate new design solutions.

The chapter presents a novel approach to producing creative designs of mechanical products during the conceptual design phase. The approach employs a knowledge base comprising rules that implement a Behaviour-Entity-Constraint (BEC) representation of modified TRIZ inventive principles. The BEC representation enables the generation of innovative solutions based on TRIZ knowledge. The objective is to retrieve existing inventive principles automatically as solutions according to design requirements.

## 4.2 Behaviour-Entity-Constraint (BEC) representation of inventive principles

### 4.2.1 Modification of TRIZ inventive principles based on Behaviour-Entity (BE) representation

Among the comprehensive array of TRIZ tools, the set of 40 Inventive Principles (IPs) is a useful creativity tool for a variety of problem solving situations. However, the IPs are often criticised for their illogical sequencing, their level of overlap, the gaps that they contain, and most of all, the difficulty people experience in remembering them all. A systematic structure of modified inventive principles has been presented in Chapter 3 to address this issue. Each principle is a method for solving problems. This normally consists of behaviour and an entity or an attribute, expressed by two sets of relations between predicates and objects respectively. The expressions are arranged in a two-dimensional matrix, the Inventive Principles Matrix (IPM), according to the predicates and the objects that they contain. This symbolic Behaviour-Entity representation is able not only to reduce the amount of repeated information in the inventive principles, but also to facilitate the evolution of current inventive principles by integrating other TRIZ or TRIZ-derived tools. Chapter 3 also presented a case study to demonstrate how the symbolic expression of inventive principles can be applied to solve problems. However, the Behaviour-Entity expression of principles only contains information on a solution. It can only be retrieved when the information of behaviour or entity is obtained, but cannot be retrieved according to the design requirements. Therefore, constraints on the solution need to be added to the BE expression to support conceptual design.

### 4.2.2 Functional objectives and contradiction constraints in BEC representation

In a TRIZ matrix of contradictions (Figure 4.1) the rows indicate system features that the designer typically wants to improve. Columns refer to undesired effects arising from improvements to a row parameter. Normally, the functional objective of a design is specified independently as a function of the design. The task is to investigate solutions that meet this individual objective. The links between objectives and principles are shown in the CM (Contradiction Matrix).

A contradiction is a pair of parameters in conflict with each other. Each matrix cell points to principles most frequently used in patents to resolve a contradiction. There are links to principles in the CM so that the principles can be retrieved according to the corresponding contradiction. Therefore, a contradiction is a constraint on a principle, called contradiction constraint. In addition, there is a second kind of constraint not involving contradictions. These constraints are functional objectives specified by the designer and represented along the first row and first column of the CM.

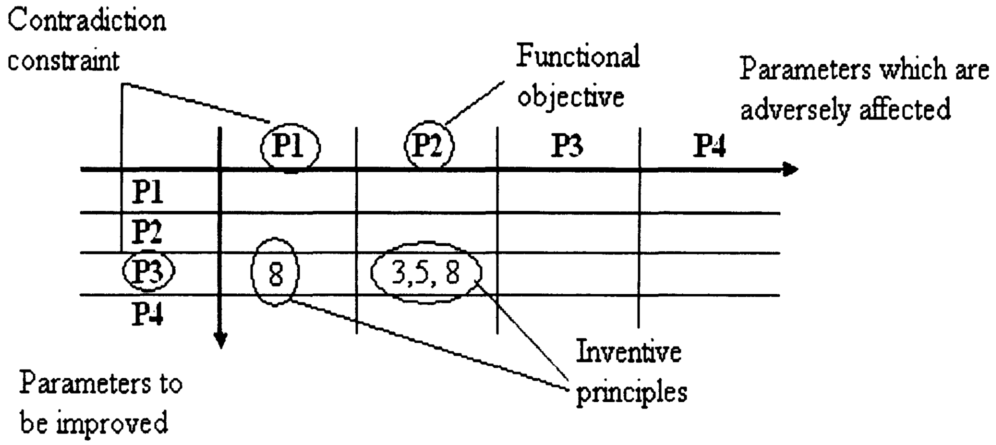


Figure 4.1 Segment of TRIZ contradiction Matrix [Altshuller 1988]

Behaviour, entity, and constraint constitute a complete representation of an inventive principle in the form of a rule in the knowledge base. The new principle contains not only information on a solution, but also information on constraints on the solution.

## 4.3 TRIZ-based design concept generation

Compared with text, the BEC representation of inventive principles offers a more precise and systematic way to retrieve information because it stores keywords of a solution without any redundant information. Boolean search is also selected as the search strategy. In chapter 3, the query based on BE representation will retrieve those principles that comprise the specified behaviour and entity. However, the behaviour within a solution/principle is hard to obtain at the beginning of design. Therefore, the query is modified based on BEC representation. A conceptual design problem is translated into a set of queries stating the contradiction constraint and corresponding entities of the design problem. The standard format of a query is $Q = (e_i, c_i)$. This query will retrieve any principles of which the symbolic expression comprises ($e_1$ OR $e_2$...OR...$e_i$) in the set of entities and ($c_1$ OR $c_2$ ...OR...$c_i$) in the set of constraints.

Figure 4.2 shows the flow of the proposed TRIZ-based design concept generation procedure. The process starts with analysing the overall functional objective of a design. That objective is then divided into several sub-objectives. Some sub-objectives may conflict with one another. These contradiction constraints or single sub-objectives together with the corresponding entities comprise the query to be presented to the TRIZ knowledge base. The knowledge base consists of BEC representations of modified TRIZ inventive principles in the form of rules. The results are those combinations of BE-represented principles that can satisfy the query. Four different situations may occur. The first two situations happen when the query can be satisfied and correspond to the two kinds of constraint. The third situation arises when constraints are not contained in the CM or corresponding principles cannot be found to satisfy constraints in that matrix. The fourth situation occurs when queries cannot be satisfied.

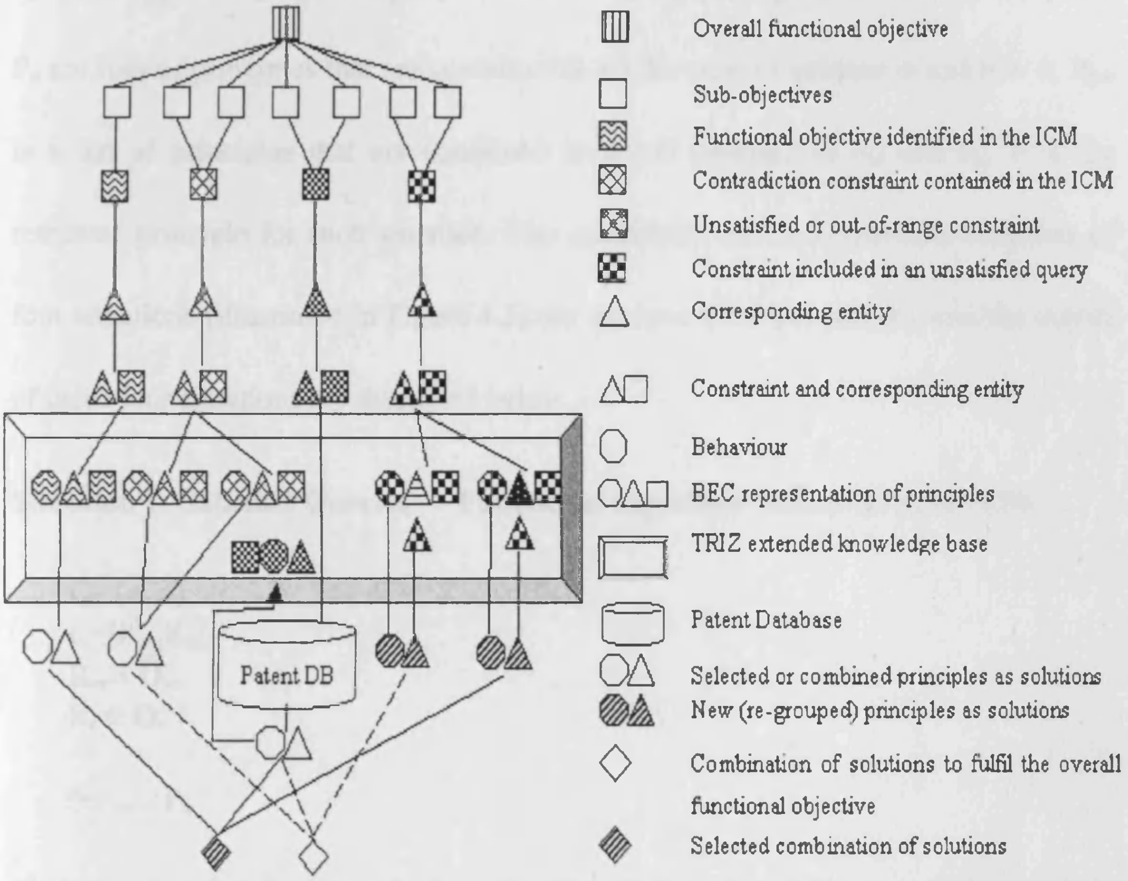

Patent DB
Overall functional objective
Sub-objectives
Functional objective identified in the ICM
Contradiction constraint contained in the ICM
Unsatisfied or out-of-range constraint
Constraint included in an unsatisfied query
Corresponding entity
Constraint and corresponding entity
Behaviour
BEC representation of principles
TRIZ extended knowledge base
Patent Database
Selected or combined principles as solutions
New (re-grouped) principles as solutions
Combination of solutions to fulfil the overall functional objective
Selected combination of solutions

Figure 4.2 Flow diagram of the TRIZ-based design concept generation process

T is the space that involves all the cells with inventive principles in CM. $R_m$ and $R_n$ are conflicting functional objectives that are identified by users/designers. $o_m \in \mathbf{O}_m$ , $o_n \in \mathbf{O}_n$. $\mathbf{O}_m$ and $\mathbf{O}_n$ separately represents row and column parameters in CM. $P_m$ and $P_n$ are lists of principles that are contained in all the cells of column m and row n. $P_{mn}$ is a list of principles that are contained in a cell pointing to $o_m$ and $o_n$. P is the retrieved principle for each situation. The conditions and corresponding solutions of four situations (illustrated in Figure 4.3) are expressed mathematically and the details of these four situations are discussed below.

**Situation 1: Satisfied Queries — Functional objectives identified in the CM.**

$$
\begin{aligned}
&Q_i=(e_i,c_i)\\
&c_i=[R_m,R_n]\\
&R_m \in \mathbf{O}_m\\
&R_n \in \mathbf{O}_n\\
&\rightarrow\\
&P=P_m \cup P_n
\end{aligned}
$$

This situation is when a sub-objective identified in the CM is specified and no negative effects can be found. In this situation, principles can be obtained from the TRIZ knowledge base according to the query. This is commonplace in conventional design because it is relatively simple for designers to work out what they wish to improve. The only problem is that many principles may be generated. The decision concerning the most suitable solution is left to the designer.

**Situation 2: Satisfied Queries — Contradiction constraints contained in the CM.**

$$\begin{aligned}&Q_i=(e_i,c_i)\\&c_i=[R_m,R_n]\in T\\&\rightarrow\\&P=P_{mn}\end{aligned}$$

This situation happens when two of the sub-objectives in the CM are in conflict. In this case, corresponding principles can be retrieved from the TRIZ database. The designer will generate the most suitable solution by selecting and combining principles from the database. This is a situation where innovative concepts will emerge.

**Situation 3: Unsatisfied or Out-of-Range Constraints.**

$$\begin{aligned}&Q_i=(e_i,c_i)\\&c_i=[R_m,R_n]\quad (c_i\notin T)\\&\rightarrow\\&P=\varnothing\end{aligned}$$

In some cases, the objective or constraint that has been specified could not be matched with any of the general parameters defined in the CM. Alternatively, certain principles used to solve design problems are not included in the CM. A blank cell in the matrix shows that there are no principles to satisfy the corresponding contradiction.

As the original CM and inventive principles were derived from the patent database, the latter could be added to the system as a secondary database. A query for this database would still be a combination of design constraints and corresponding entities. The result would be a patent that might offer a suitable solution to the design problem. This new BE-represented solution and the original contradiction constraint form a rule that can be stored into the TRIZ knowledge base for future use.

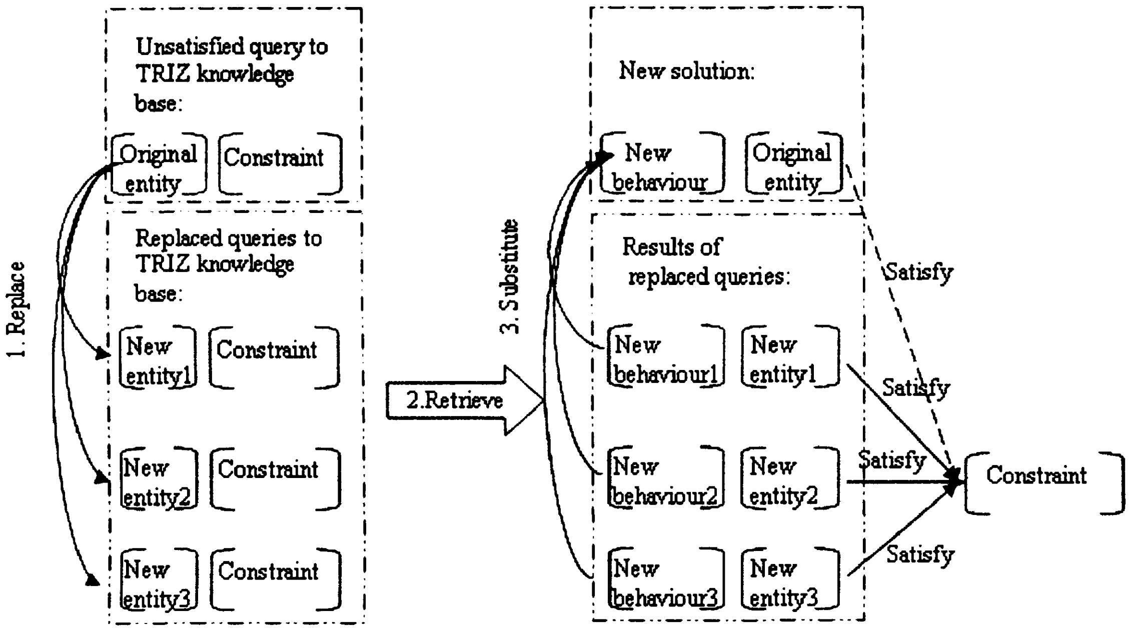


Figure 4.3 Generation of new design solutions based on BEC representation

**Situation 4: Unsatisfied Queries.**

In this case, there is no matching principle that can be found to satisfy both the constraints and entities at the same time. The new TRIZ-based approach is applied to generate feasible solutions by re-organising the three elements in the BEC representation (Figure 4.3). The hierarchical structure of the list of entities has been explained in [Pham 2006], **P** is a list of all the inventive principles in CM.

$$Q_i=(e_i,c_i)$$
$$c_i=[R_m,R_n] \in T$$
$$P_j=(b_j,e_j,c_j) \in \mathbf{P}$$
$$(e_i,c_i) \cap (e_j,c_j) = \emptyset$$
$$\rightarrow$$
$$P=P_{new} \quad ( P \notin \mathbf{P} )$$

If a query is not satisfied, each original unmatched entity is replaced with its neighbour (entity at the same level) or its parent (entity at a higher level) until a matching solution is found. The aim of this step is to find similar solutions that could resolve the same contradiction. Combining a new behaviour with the original entity may make up an effective solution to the required constraint. Like many other creative thinking techniques, this process may not be sufficient for completely solving a design problem, but it does help the designer to generate new potential solutions.

The final solution to the given design problem is obtained by combining the solutions generated in all the situations applicable to the problem.

## 4.4 Case study

The contradiction between "usage of area of floor" and "volume of fuselage" is a typical problem in fuselage design. It has been selected in this example to show how it can be solved. Carter [2001] has demonstrated how to solve this contradiction by applying TRIZ contradiction matrix and inventive principles. The section will then present how to solve the same contradiction by the proposed TRIZ-based design approach, and the advantages and different results obtained from applying the new approach will be given for comparison to classical TRIZ.

### 4.4.1 Case 1: Classical TRIZ

In Carter's case study, he first analyzed a pair of conflict objectives "usage of area of floor" and "volume of fuselage". The corresponding parameters in the contradiction matrix are then identified as "area of moving object" and "volume of moving object".

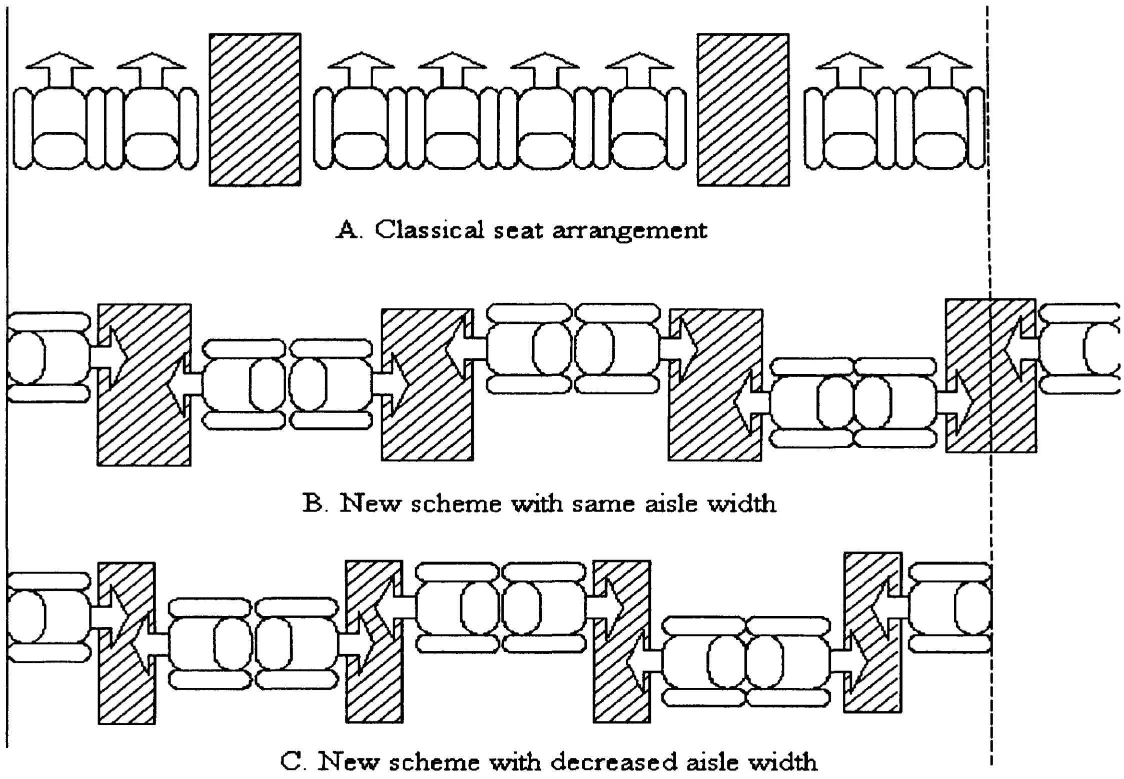
The contradiction matrix provides the following principles (Appendix A) as suggested solutions

- P7. Nested Doll
- P14. Spheroidality
- P17. Another dimension
- P4. Asymmetry

Carter gives interpretations of suggested inventive principles above in his report of the case study *the application of TRIZ to economy class aircraft cabin design.*

The starting point is a standard seat layout with all seats arranged in rows facing forwards. Introducing asymmetry is beneficial since facing passengers can "share" legroom and have more space. The gap in front of each seat and the aisle can be combined by seating the passengers sideways. Unfortunately, anybody using the aisle would have to pick his or her way through a long row of legs. This may be solved by re-applying the "another dimension" trigger and raising the aisle. The legs of passengers are accommodated underneath ('within') the aisle.

This scheme has advantages. Usage of floor area has been significantly improved in two respects without increasing the cabin capacity: legroom has greatly increased so has the number of aisles. However, a "side-effect" has resulted: the width of aisle has decreased significantly (by 50%), or the number of seats decreases, as can be seen from Figure 4.4.



A. Classical seat arrangement

B. New scheme with same aisle width

C. New scheme with decreased aisle width

Figure 4.4 Comparison of aisle width between classical seat arrangement and proposed seat arrangement

### 4.4.2 Case 2: Improved TRIZ

The same contradiction needs to be resolved, represented as $C_5$ (Area of moving object ) and $C_7$ (Volume of moving object). The entity is identified as "area", which is represented as y5aaac. The query that needs to be satisfied is expressed as ($C_5$ Vs $C_7$) and (y5aaac).

Although the principles to resolve ($C_5$ Vs $C_7$) can be retrieved, the complete expression ($C_5$ Vs $C_7$) and (y5aaac) cannot be satisfied. Then the approach in situation 4 will be applied. The present entity y5aaac is replaced with its same level (i.e. y5aaab), its super level (i.e. y5aaa), or its super level's neighbour's sub level (i.e. y5aaba) and the corresponding principles are retrieved. Finally, the new entities in these retrieved principles are replaced by the original entity again to make up the new solutions. The results of this process are shown in Table 4.1.

| Inputs | Retrieved principles | New solutions |
|---|---|---|
| ($C_5$ Vs $C_7$) and (y5aaac) | None | |
| ($C_5$ Vs $C_7$) and (y5aaaa) | None | |
| ($C_5$ Vs $C_7$) and (y5aaab) | P17E: ***[x1c]([x2a][y5aaab<y4])***<br>P17D:[x8a][y5aaab<y4]<br>***P17C:[x2a][y5aaab]*** | [x1c]([x2a][y5aaac<y4])<br>***[x8a][y5aaac<y4]***<br>[x2a][y5aaac] |
| ($C_5$ Vs $C_7$) and (y5aaa) | None | |
| ($C_5$ Vs $C_7$) and (y5aab) | P4B:[x4c][y5aab]←[y5aaba<y3] | [x4c][y5aaac]←[y5aaba<y3] |
| ($C_5$ Vs $C_7$) and (y5aaba) | P4C:[x2a][y5aaba]<br>P4B:[x4c][y5aab]← [y5aaba<y3]<br>***P4A:[x5c][y5aaba]*** | [x2a][y5aaac]<br>[x4c][y5aab] ← [y5aaac<y3]<br>[x5c][y5aaac] |
| ($C_5$ Vs $C_7$) and (y5aabb) | P14A:[x5c][y5aabb] | [x5c][y5aaac] |

Table 4.1 Input and output of solution generation process

The results generated by this new approach include more specific principles, which will give a clearer solution to the given problem.

In figure 4.5 (a), the starting point is also a standard seat layout with all seats arranged in rows facing forwards. The new solution [x8a][y5aaac<y4] means "move the area". The area in this case indicates the aisle area or the legroom of passengers, which is shown in grey in Figure 4.5 (a).

[x1c]([x2a][y5aaab<y4]) means "make use of another side of a given object". If the passenger seats are considered as an object, then the combination of these two solutions can result in a new solution: move the area (legroom) to another side of the passenger seats. As shown in Figure 4.5 (b), passengers face outward with seating back to back (b).

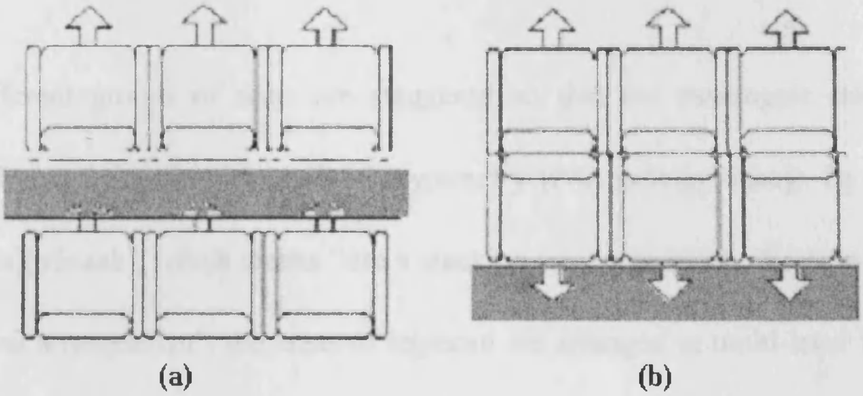


Figure 4.5 Application of P17 and the re-organized principle

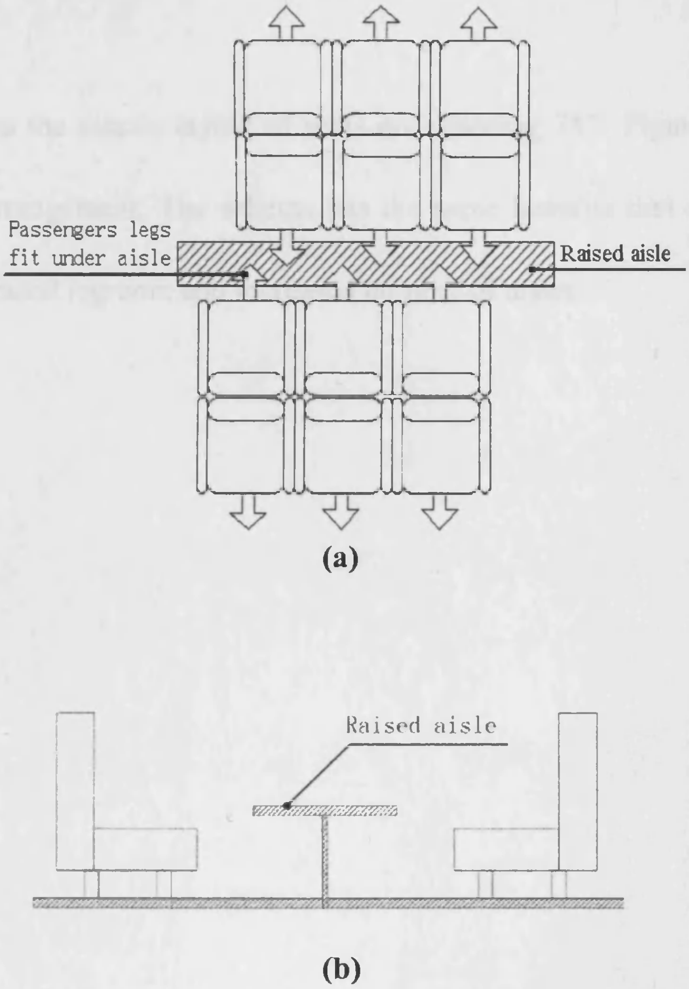


Figure 4.6 Applications of P4

Next, different groups of seats are staggered so that the passengers can "share" legroom by applying the principle of asymmetry (P4A:[x5c][y5aaba]). By applying P17C:[x2a][y5aaab], which means "use a stacking arrangement of objects instead of a single level arrangement", the areas of legroom are arranged in multi-level instead of a single level. As shown in Figure 4.6, passengers' legs are accommodated underneath ('within') the aisle.

Figure 4.7 shows the classic layout of seats on a Boeing 747. Figure 4.8 shows the modified seat arrangement. The scheme has the same benefits that case 1 provides, such as the increased legroom and increased number of aisles.

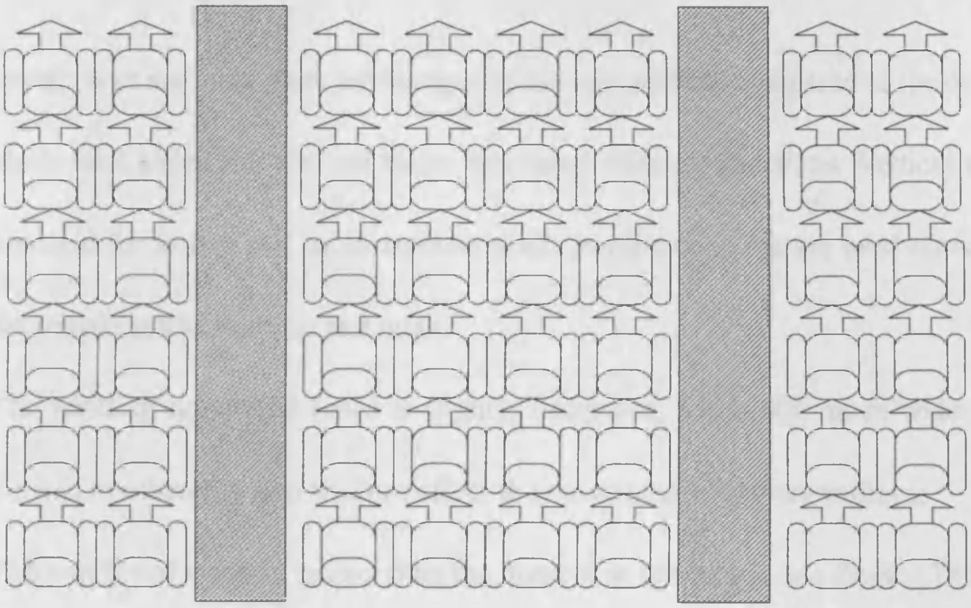


Figure 4.7 Classic layout

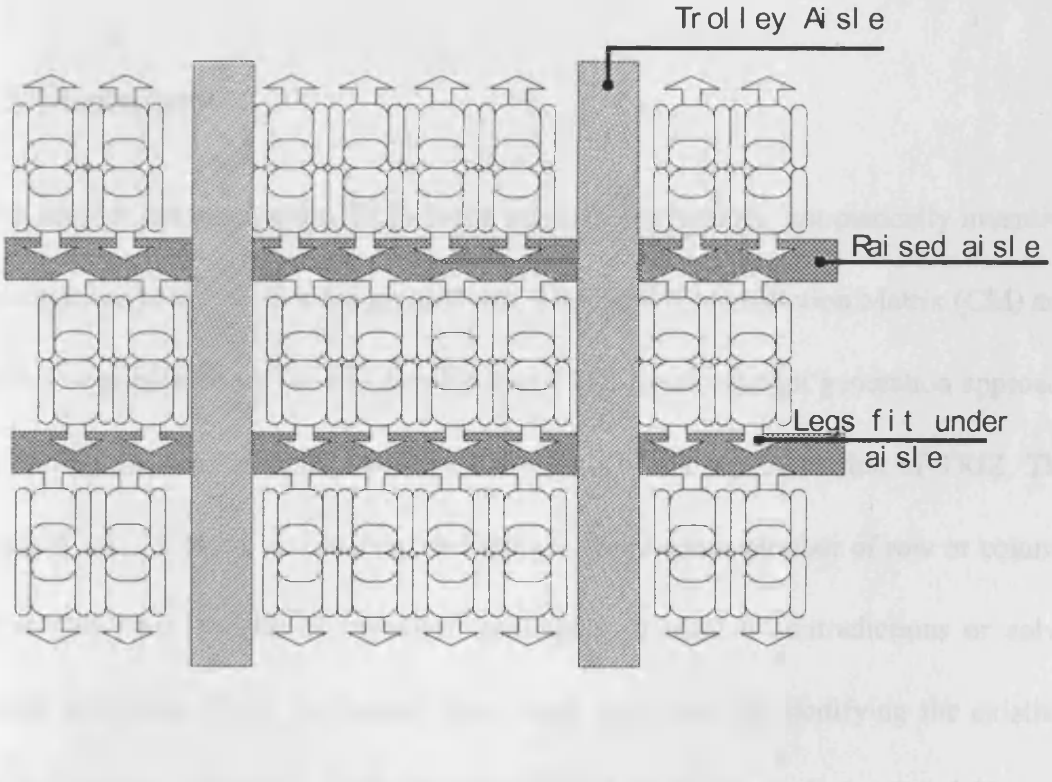


Figure 4.8 Modified layout

Moreover, there are three more advantages of the new scheme compared to the case 1:

- Horizontal aisles and vertical aisles can carry different functions. Vertical aisles are used for trolley and cabin service, while horizontal aisles are used especially for passengers to exercise and relax.
- The width of horizontal aisles is slightly decreased, but is still much wider than the one in scheme 1, and will not affect the movement of service trolleys.
- The number of aisles is greater than the number in scheme 1. In a Boeing 747-400 economy class cabin, there are 32 rows of seats. Therefore, 15 aisles will be added in scheme 2 rather than the 2 aisles added in scheme 1, which will provide increased space for passengers.

## 4.5 Summary

This chapter has proposed a TRIZ-based approach to retrieve automatically inventive principles as solutions to a design problem. The TRIZ Contradiction Matrix (CM) and inventive principles are used to develop this TRIZ-based concept generation approach by adding constraints to the standard Behaviour-Entity representation of TRIZ. The original TRIZ CM has limitations, including its inadequate number of row or column parameters and its lack of inventive principles to resolve contradictions or solve design problems. These limitations have been overcome by modifying the existing TRIZ inventive principles, and applying the CM as the core of an expandable and customised TRIZ knowledge base. Moreover, the layout design of the fuselage of an aircraft has been used to show that the TRIZ-based design methodology is more efficient than classical TRIZ, which is the second hypothesis presented in Chapter 1.

# Chapter 5. Knowledge-based conflict resolution

## 5.1 Preliminaries

Design is a problem solving activity based on multiple and diverse sources of expertise. According to Klein [2000], a conflict is an incompatibility between two (or more) decisions concerning the design or the designer's goals. It has also been stated that a conflict is linked to a divergence of goals [Castelfranchi 2000]. In a concurrent engineering (CE) environment, conflict becomes much more explicit in that all members involved in a design project are expected to join in the discussion and negotiation over any disagreements encountered. Negotiation is a widely accepted approach for conflict resolution in collaborative design. However, negotiation is an ineffective approach for conflict resolution in that one side is normally obliged to abandon its goal partially or fully. The two sides of a conflict can only be partially satisfied even through win-win negotiation. Negotiation is particularly ineffective when the two sides are in dispute. This chapter presents a method that integrates a "no-compromise" approach using TRIZ [Altshuller 1988] and a negotiation-based multi-stage approach to support conflict resolution in collaborative conceptual design.

The chapter describes the classification, detection, and resolution of three types of conflict in a collaborative design environment. Special attention is paid to the resolution of the third type of conflict. Finally, this chapter details the implementation of the proposed conflict resolution method using the rule-based language JESS.

## 5.2 Conflict classification and detection in design

A TRIZ-based approach for conflict resolution and concept generation to assist individual designers has been presented in the previous chapters. The term "conflict" is defined here as a situation of conflicting design goals, also called contradiction constraints in chapter 4. Three types of conflict may occur in a design system when a group of designers are working in a collaborative environment. They are represented as Conflicts A, B, and C in Figure 5.1.

The collaborative design process starts with analysing an individual design objective/goal and identifying its corresponding TRIZ contradiction matrix parameter. The list of conflicts is determined by information on goals identified by designers. A model of a design goal has been developed to enable the process of automatic conflict detection.

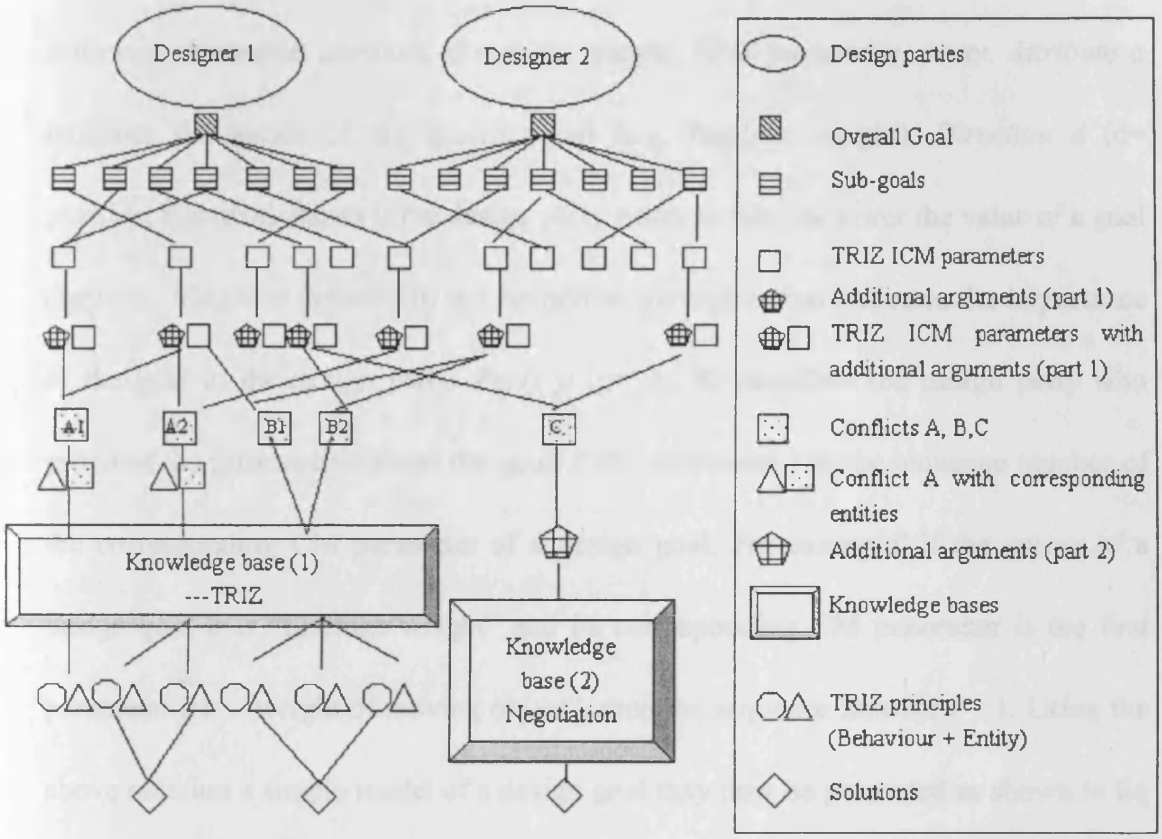

Designer 1
Designer 2
A1
A2
B1
B2
C
Knowledge base (1)
---TRIZ
Knowledge base (2)
Negotiation
Design parties
Overall Goal
Sub-goals
TRIZ ICM parameters
Additional arguments (part 1)
TRIZ ICM parameters with additional arguments (part 1)
Conflicts A, B,C
Conflict A with corresponding entities
Additional arguments (part 2)
Knowledge bases
TRIZ principles (Behaviour + Entity)
Solutions

Figure 5.1 Three types of conflict and conflict resolution process

In the proposed system, two-party collaboration is considered, with the parties denoted as A and B. This research concentrates on the qualitative aspects of conceptual design. A qualitative structure of a design goal is described using the following attributes: *attribute, direction, weight, TRIZ parameter, party*. *Attribute a* indicates the nature of the specific goal (e.g. fuselage weight). *Direction d* (d= positive, negative) shows if the design party wants to raise or lower the value of a goal attribute. *Weight w* ($0<w\leqslant 10$) is a numerical parameter that indicates the importance of the goal to the design party. *Party p* (p= A, B) describes the design party who provides the information about the goal. *TRIZ parameter t* is the sequence number of the corresponding CM parameter of a design goal. For example, if the nature of a design goal *a* is "fuselage weight" and its corresponding CM parameter is the first parameter P1= "weight of moving object", then the sequence number $t = 1$. Using the above notation a simple model of a design goal may now be presented as shown in Eq 5.1.

$$Gi=G\ (ai,\ di,\ wi,\ ti,\ pi\ ) \qquad 5.1$$

Let T be the space that comprises all the cells containing inventive principles in CM. Each cell in CM is defined by **t**, $\mathbf{t}=[t_x,t_y]\in T$. $t_x$ and $t_y$ are the sequence numbers of the system parameters in the first row and first column that point to cell **t**. The space R consists of all the design goals $G_i$. $G_i= G\ (a_i,\ d_i,\ w_i,\ t_i,\ p_i\ ) \in R$. The three lists of conflicts and their detection rules are described below.

**Conflict A:** Significant conflict. This type of conflict comes from either an individual designer or from different designers, and is shown as A1 and A2 in Figure 5.1. Conflict type A is a conflict that is regarded as important/significant by designers. The weights given by designers for their conflicting goals should be close to the maximum value 10. (For example, $w_i$>9.5 and $w_j$ >9.5 at the same time). The products of $w_i$ and $w_j$ should therefore exceed a minimum threshold, chosen as 90 in this case. The previous chapter discussed this type of conflict and the method of resolving it by searching TRIZ CM. Therefore, one of the conditions for identifying conflict A is that the two TRIZ parameters involved in the conflicting goals should be included in CM. At the same time, these two parameters should point to a non-blank cell. The above conditions are expressed mathematically as follows.

**Rule for detecting conflict A:**
$\forall G(w_i, t_i) \in R$
$\forall G(w_j, t_j) \in R$
$[t_i,t_j] \in T$
$(w_i \times w_j) \geqslant c$ $(c=90)$
$\rightarrow$
Conflict A

**Conflict B:** Multiple conflicts. For a type A conflict, solutions are retrieved and generated for each conflict. However, more than one pair of conflicting requirements is identified in most cases in engineering design. The resolutions retrieved for each conflict by searching the TRIZ knowledge base may result in new conflicts in this situation. In order to avoid this problem, type B conflicts are solved in a different way. Solutions for type B conflicts normally address multi-conflicts all at once instead of solving individual conflicts as in type A. The resolution of conflict B will also be manipulated by searching TRIZ CM. Therefore, the basic condition for detecting conflict B is the same as that for conflict A. The difference between conflict A and conflict B is that conflict B is not considered as significant as conflict A by designers. The product of the goal weights should not be more than c in this situation.

**Rule for detecting conflict B:**
$\forall G(w_i, t_i) \in R$
$\forall G(w_j, t_j) \in R$
$[t_i,t_j] \in T$
$(w_i \times w_j) < c$ (c=90)
→
Conflict B

**Conflict C:** Negotiable conflicts. In concurrent design, each design party has its local goals. One TRIZ parameter may be related to multiple specific goals. In some cases, two design parties may want to achieve the different specific goals with the same TRIZ parameter. For example, in an aircraft design, designer A is responsible for designing the cargo area, and wants to improve the cargo space. Designer B is responsible for cabin design, and wants to increase cabin space. It can be seen that both of their goals are connected to the same TRIZ parameter No.7 " volume of moving object".

This type of conflict is shown as a blank cell in the diagonal of CM. Therefore, the matrix is unable to offer principles or solutions to this type of conflict. Unlike conflicts A and B, conflict C is resolved by a negotiation-based multi-stage approach instead of searching TRIZ CM.

The basic condition for identifying conflict C is that the TRIZ parameter related to two conflicting goals in this type of conflict should be the same. Moreover, negotiation takes place between at least two participants. Thus, these two parameters should be from different design parties.

**Rule for detecting conflict C:**
$\forall G(t_i, p_i) \in R$
$\forall G(t_j, p_j) \in R$
$(t_i=t_j)$
$(p_i=A)$
$(p_j=B)$
→
Conflict C

# 5.3 Conflict resolution

## 5.3.1 Conflict A and B resolution

Conflict A is solved one by one by making a query to the TRIZ knowledge base. The standard format of a query is $Q_i=(e_i, c_i)$, where $e_{i,}$ and $c_i$ are the entity and constraint of a design problem. However, if conflict B is solved individually, as in the case of conflict A, the principles generated may result in new conflicts. When the complexity of the design increases, the efficiency of the system will decrease. In order to eliminate this problem, the query is modified as $Q= (c_1, c_2 ......c_i)$. The entity element $e_i$ is used in the query of conflict type A to generate new solutions by re-organising principles. It is removed in the case of conflict type B because the main task here is to retrieve principles. This task can be achieved by using one attribute $c_i$ in the modified query. This new query will retrieve principles that satisfy as many of the constraints in the set ($c_1$, $c_2$ ......$c_i$) as possible.

The ideal result comprises principles that can satisfy all the constraints identified by designers. The approaches to resolving conflict A and conflict B have been detailed in chapter 3 and chapter 4.

### 5.3.2 Conflict C resolution

Conflict C is composed of two goals with the same TRIZ parameter from two design parties. This conflict can be divided into three categories according to the nature of the goals. The first category involves the same global or local goal but with different directions. For example, designer A wants to increase the weight of an object, while designer B wants to decrease it. The second category involves different local goals with the same direction. For example, designer A wants to increase the size of the cabin, while designer B wants to increase the size of the cargo area. The third category involves different goals (one global, one local) with different directions. For instance, designer A wants to increase the weight of the cabin, while designer B wants to decrease the weight of the fuselage. The different conflict C categories can be solved by corresponding resolution strategies. Figure 5.2 shows the taxonomy of resolution strategies, the conditions of the strategies, and the connections between them.

The taxonomy of resolution strategies was inspired by the work of Mark [1992], although there was no priority for the strategies in Mark's work. The modification was made by arranging strategies from left to right according to their originality, validity, and ease of implementation. For example, once the condition of the strategy "abandon goal" is satisfied, it is an easy operation for a design party to give up the design goal. However, this operation might lead to violation of this design party's main objective/goal because of the abandoning of this specific goal. On the other hand, if "an alternate plan" can be found, each involved side does not need to compromise too much. Thus this strategy is given a higher priority than the previous one.

At the same time, the strategies for each category of conflict C will be retrieved according to different conditions (or constraints). These conditions are basically suggested by four different parties: the designers, the client, other related design parties, and the authorities. They are arranged from bottom to top according to their relevance. Designers are at the bottom, which means the conditions set by them are regarded as being most relevant to the problem. On the other hand, the opinion of the authorities is not considered until the conflict cannot be resolved by the other three parties.

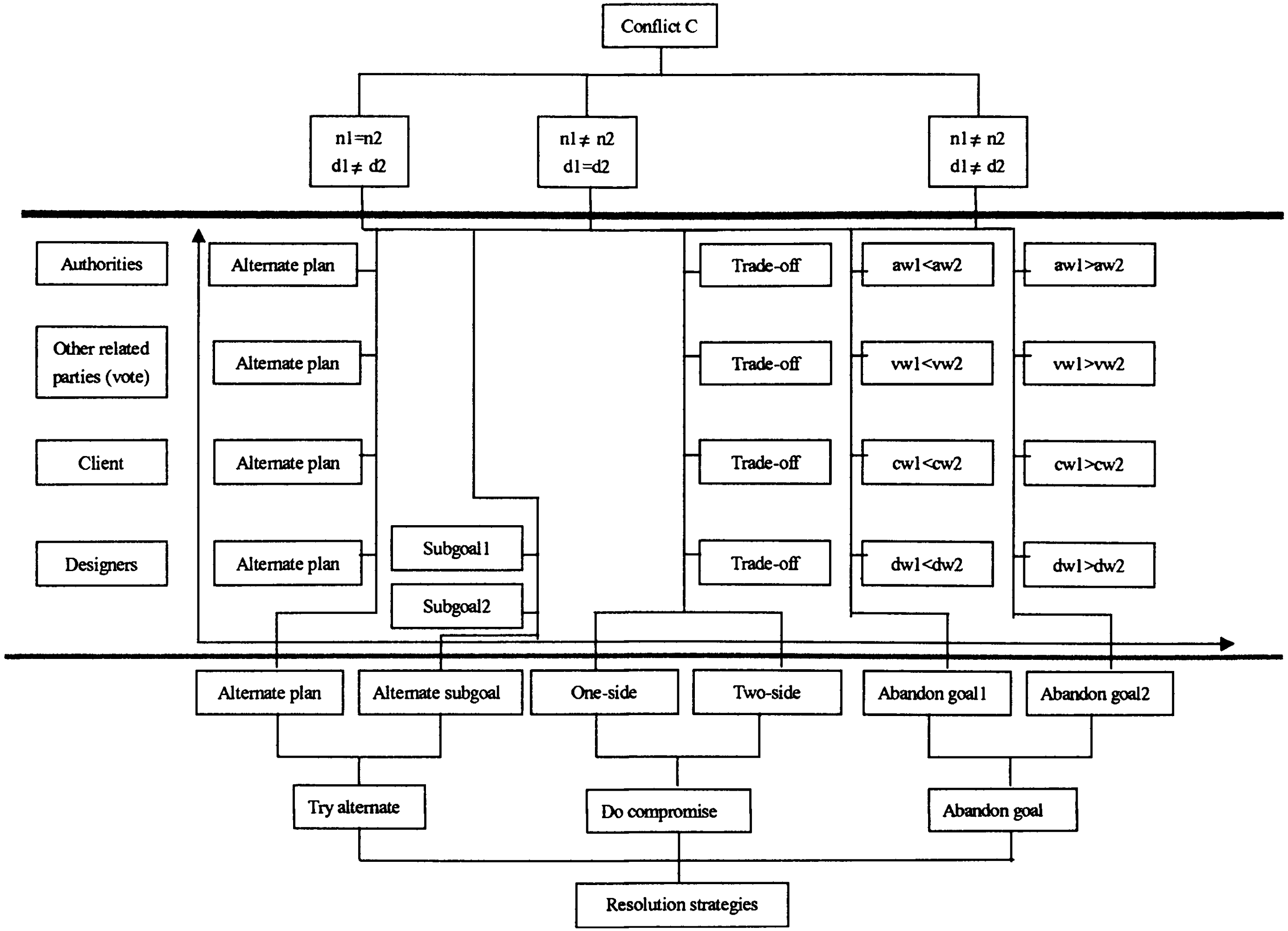


Figure 5.2 Taxonomy of conflict C

In order to find corresponding resolution strategies, more properties/attributes of design goals are needed: *global/local, sub-goal, client-weight, vote-weight, authority-weight.* Together with the attributes defined in the last section, the model of a design goal is expanded and modified as Eq 5.2 and Eq 5.3:

$$\mathrm{G_i}{=}\mathrm{G}\ (n_i,\ d_{i,},\ t_{i,}\ ,p_{i,},\ gl_i\ sg_{i,},\ w_{mi\,,}\ ) \qquad 5.2$$

$$\mathrm{G_i}{=}\mathrm{G}\ (n_j,\ d_j,\ t_{j,}\ ,p_{j,},\ gl_{j,}\ sg_{j\,,}\ w_{mj\,,}\ ) \qquad 5.3$$

$$alt_k = alt\ (ap_{mk},\ an_k) \qquad 5.4$$

$$com_k = com\ (cp_{mk}, cn_k) \qquad 5.5$$

$$\mathrm{C_k}{=}\mathrm{C}\ (\mathrm{G_i}, \mathrm{G_j},\ alt_{k,}\ com_{k,}\ ) \in \mathrm{U} \qquad 5.6$$

Where *gl*= (Global, Local) shows if a design goal is a global or local attribute; *sg* indicates the name of a sub-goal; $w_m$ indicates the importance of a goal appointed by different parties; m=1, 2, 3, 4, and is a pointer which identifies the 4 involved parties.

Eq 5.6 shows a model of a conflict C, which is composed of two design goals. Two attributes, *alt* and *com*, are added to the model of C to enable the search for resolution strategies. Both *alt* and *com* have two sub-attributes: name and party, which separately show the name of a plan and that of the party who provides an alternate or compromise plan (see Eq 5.4 and Eq 5.5). The above information is to be acquired from the four parties. However, they do not have to provide answers for all these fields, although the more knowledge of design goals is offered, the more precise the result will be.

The conditions of each resolution strategy are shown in Table 5.1. The differing relevance of the four parties is indicated by the rate $vp_m$, which is set for each party for calculating the weight of each resolution strategy as: $vp_1$=1, $vp_2$ = 0.8, $vp_3$ = 0.6, and $vp_4$ =0.4. As each resolution strategy also has a different priority, the priority for them is set as: $vs_1$ = 1, $vs_2$= 0.8, and $vs_3$ = 0.6. Here, indices 1, 2, and 3 respectively indicate resolution strategies “Try alternate”, “Do compromise”, and “Abandon goal”. The calculation of the weight for each strategy is also described in Table 5.1.

| Conditions | Resolution strategies | |
|---|---|---|
| | Calculation of weight | Name |
| $\exists\, alt_{k,}\ ((ap_{mk}, an_k) \in C_{k,}$ | $ap_{mk} \times vp_m \times vs_1$ | Alternate plan |
| $\exists\, sg_{i,} \in G_{i,}\ \vee\ \exists\, sg_{i,} \in G_{i,}$ | $vp_1 \times vs_1$ | Alternate sub-goal |
| $\exists\, com_{k,}\ ((cp_{mk}, cn_k) \in C_{k,}$ | $cp_{mk} \times vp_m \times vs_2$ | Compromise |
| $\exists\, w_{mi,} \in G_{i,}\ \wedge\ \exists\, w_{mj} \in G_{i,}$<br>$\Sigma(w_{mi} \times vp_m) < \Sigma(w_{mj} \times vp_m)$ | $\dfrac{\sum(\mathrm{wmj} \times vpm) - \sum(\mathrm{wmi} \times \mathrm{vpm})}{(\sum(wmi \times vpm)) \times vs3}$ | Abandon Goal i |
| $\exists\, w_{mi,} \in G_{i,}\ \wedge\ \exists\, w_{mj} \in G_{i,}$<br>$\Sigma(w_{mi} \times vp_m) > \Sigma(w_{mj} \times vp_m)$ | $\dfrac{\sum(\mathrm{wmi} \times vpm) - \sum(\mathrm{wmj} \times \mathrm{vpm})}{(\sum(wmj \times vpm)) \times vs3}$ | Abandon Goal j |

Table 5.1 Conditions of resolution strategies

## 5.4 Implementation

### 5.4.1 The implementation language used: JESS

JESS (Java Expert System Shell) is a Java derivative or clone of the popular expert system shell CLIPS (C Language Integrated Production System) designed by researchers at NASA. JESS /CLIPS is a well-known expert system language used in many systems and as a result has a wide programmer base. JESS has the capacity to "reason" using knowledge expressed in the form of declarative rules. Therefore, a series of rules for searching, retrieving, and generating solutions can be built using JESS.

There are a few rule engines available. JESS was selected to implement the approach because it is small and does not require much computing resource. It is also a fast engine which uses an enhanced version of the Rete algorithm to process rules. Its powerful scripting language gives the developer access to all of Java's APIs (Application Programming Interfaces).

### 5.4.2 System architecture

Figure 5.3 shows the architecture of the implemented system. The system including the GUI was developed using only JESS scripts, and JESS's Java reflection capabilities, in particular, Java's Swing library. User requirements are input through the GUI, and stored as facts in the working memory. The working memory in this system consists of two main parts. The first part stores user-input information transferred from the GUI. The second part comprises TRIZ-related facts and negotiation facts. TRIZ inventive principles and CM parameters are stored here. At the same time, resolution strategies based on negotiation are also stored as facts in working memory.

Another important component in this system is the rule base, which stores rules for detection and resolution of conflicts. These rules are divided into four modules and will be fired one module after another. Once the inference engine starts, all the rules are compared to the working memory (using the pattern matcher) to decide which ones should be activated during this cycle. All the activated rules are ordered to form the agenda, which will help to decide which rule to fire first. Finally, the execution engine will execute the action part of the selected rule and obtain the output of the system. The latter will be transmitted to the user though the GUI. System outputs may include data requests of the user, in which case the system will wait until the requested data is provided. The entire process is repeated until the final output is obtained and the inference engine stops.

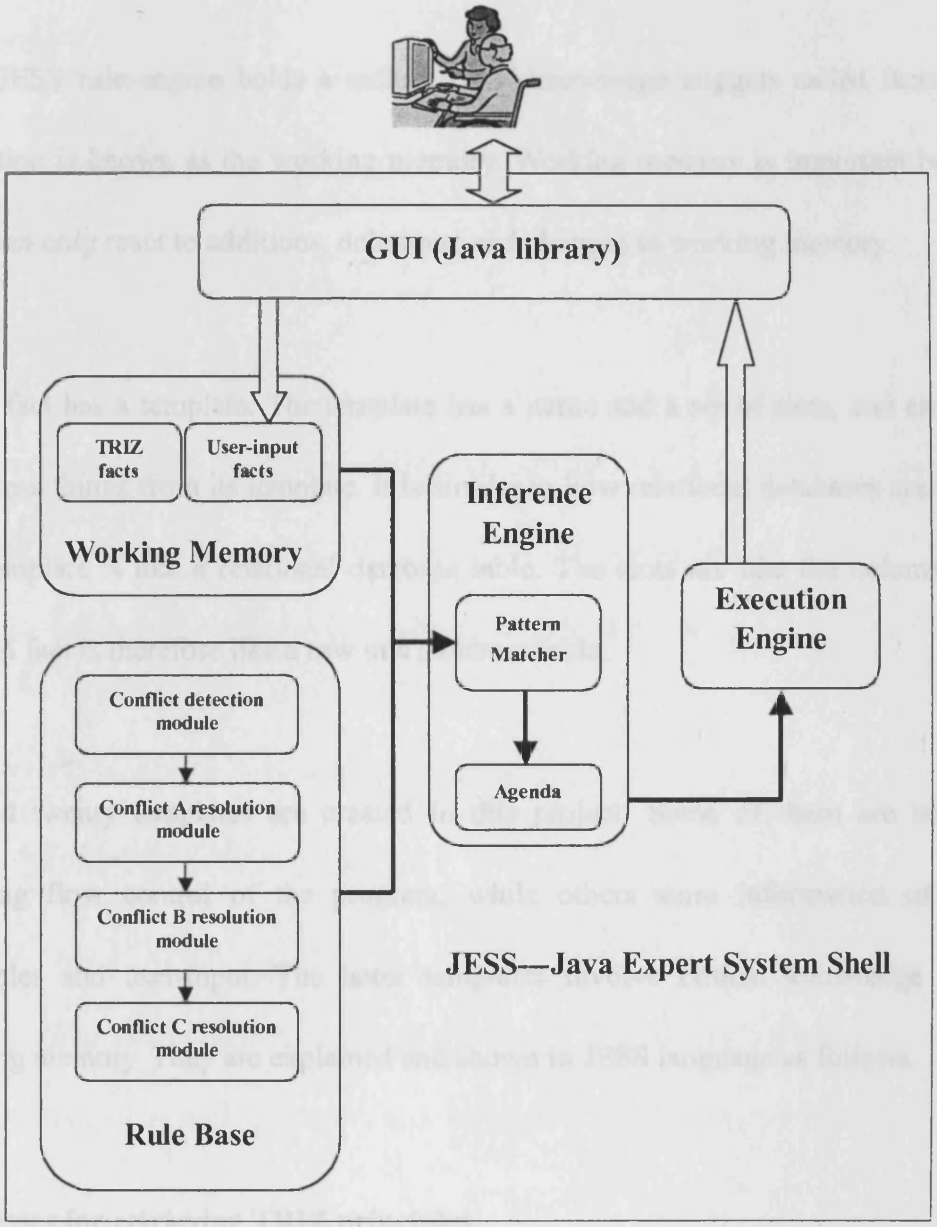

GUI (Java library)
TRIZ facts
User-input facts
Working Memory
Inference Engine
Pattern Matcher
Agenda
Execution Engine
Conflict detection module
Conflict A resolution module
Conflict B resolution module
Conflict C resolution module
Rule Base
JESS—Java Expert System Shell

Figure 5.3 System architecture

### 5.4.3 Working memory

Each JESS rule engine holds a collection of knowledge nuggets called facts. This collection is known as the working memory. Working memory is important because rules can only react to additions, deletions, and changes to working memory.

Every fact has a template. The template has a name and a set of slots, and each fact gets these things from its template. It is similar to how relational databases are set up. The template is like a relational database table. The slots are like the columns of a table. A fact is therefore like a row in a database table.

Around twenty templates are created in this project. Some of them are built for assisting flow control of the program, while others store information of TRIZ principles and user-input. The latter templates involve critical knowledge in the working memory. They are explained and shown in JESS language as follows.

**Templates for retrieving TRIZ principles**

Information on TRIZ system parameters is stored in the template ***objectives***. These parameters appear on the first column and first row of the TRIZ contradiction matrix. Slot *name* and *number* indicate the sequence number and name of the parameters. Slot *catename* and *cateno* are the name and sequence number for the category of the objective.

```
 (deftemplate objectives
        "functional objectives"
   (slot number)
   (slot catename)
   (slot cateno)
   (slot name))
```

The template ***final-contradiction-constraint-with-cell*** stores the sequence number of a pair of the contradictive objectives, and their corresponding cell in the TRIZ CM, including a list of sequence numbers of principles in this cell.

```
(deftemplate final-contradiction-constraint-with-cell
   (slot first-objective)
   (slot second-objective)
   (multislot cell))
```

The next important template is called ***principles***. It stores information of sequence number and text of each principle, and the symbolic expressions of each of its sub-principles.

```
(deftemplate principles
   (slot number)
   (slot text)
   (slot symbol-expression1)
   (slot symbol-expression2)
   (slot symbol-expression3)
   (slot symbol-expression4)
   (slot symbol-expression5))
```

## Templates for translation and reorganisation of inventive principles

The symbolic expression of each principle is stored in the template ***principles-for-translation***. As mentioned in chapter 3, there are 12 forms of the symbolic expressions. Each form has its own translation rule. The form of expression is determined by the value of slots in this template.

```
(deftemplate principles-for-translation
   (slot number)
   (slot supernumber)
   (slot expression)
   (slot text)
   (slot first-priority-x1)
   (slot first-priority-x2)
   (slot first-priority-xr)
   (slot first-priority-y1)
   (slot first-priority-y2)
   (slot first-priority-yr)
   (slot x1)
   (slot x2)
   (slot x3)
   (slot xr1)
   (slot xr2)
   (slot xrf)
   (slot y1)
   (slot y2)
   (slot y3)
   (slot yr1)
   (slot yr2)
   (slot yrf)
   (slot so-symbol)
   (slot ex1)
   (slot ex2)
    )
```

## User-input template

The template ***Goal*** includes information of goal, which is acquired from the user. This template is used for detecting the catalogue of conflict. The next template ***goal-extra*** stores information that is input by the user as well. It is built for resolving conflict type C.

```
(deftemplate Goal
   (slot name)
   (slot direction)
   (slot weight)
   (slot T-parameter)
   (slot party))

(deftemplate goal-extra
   (slot name)
   (slot T-parameter)
   (slot Local/global)
   (slot designerweight)
   (slot clientweight)
   (slot voteweight)
   (slot authorityweight)
   (slot subgoal)
   (slot catelogue))
```

### 5.4.4 Rule base

JESS production rules consist of conditional statements known as production rules and a working memory. Contained in these production rules are one or more conditions, which lead to one or more actions. The JESS runtime cycles through the working memory trying to match conditions on the LHS with data. If it finds a rule that has enough information to fire then it places those production rules and the conditions into a conflict set and executes the relevant actions which may change this conflict set, causing more rules to fire. This cycle continues until there are no more conflicts left.

There are two types of rules in this TRIZ-based design system. The first type are *core rules* which involve key information that will lead to appropriate results of conflict detection and conflict resolution. The other type of rules are *routine rules* that are used to control flow of the program. The system consists of four modules: detection module, conflict A resolution module, conflict B resolution module, and conflict C resolution module. The flowchart of the system is shown in Figure 5.4. Both of these two types of rules will be described by different modules in following sections. The focus will be made on the core rules.

#### 5.4.4.1 Detection Module

Detection module is the first module of the program. It can also be regarded as the main module. As shown in figure 5.4, three catagories of conflicts are detected and resolved one by one. Therefore, the other three resolutions modules are sub-modules of the system.

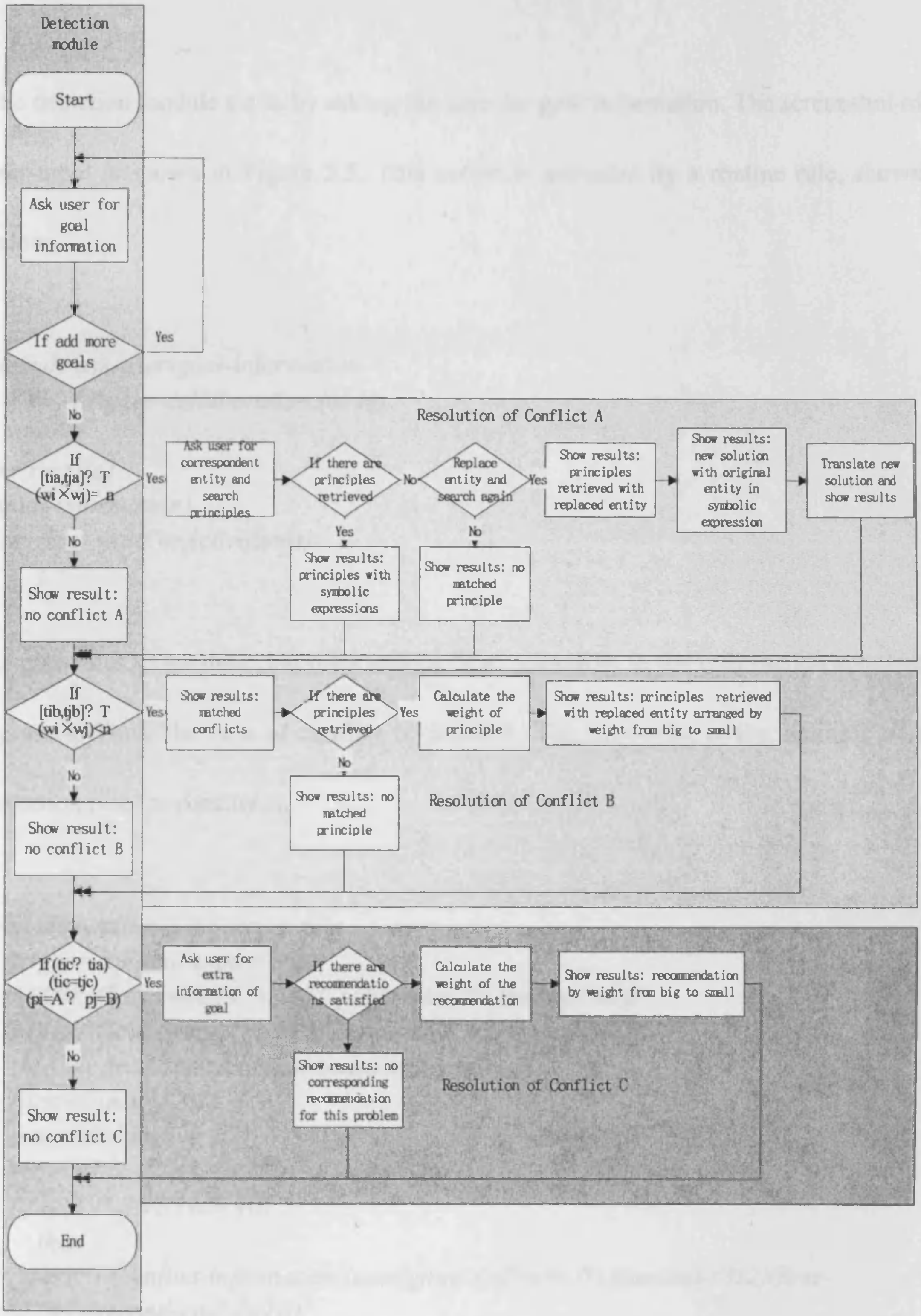

Detection module
Start
Ask user for goal information
If add more goals
Yes
No
If [tia,tja]? T (wi×wj)= n
Yes
No
Show result: no conflict A
Resolution of Conflict A
Ask user for correspondent entity and search principles
If there are principles retrieved
No
Yes
Replace entity and search again
Yes
No
Show results: principles retrieved with replaced entity
Show results: new solution with original entity in symbolic expression
Translate new solution and show results
Show results: principles with symbolic expressions
Show results: no matched principle
If [tib,tjb]? T (wi×wj)<n
Yes
No
Show result: no conflict B
Show results: matched conflicts
If there are principles retrieved
Yes
No
Calculate the weight of principle
Show results: principles retrieved with replaced entity arranged by weight from big to small
Show results: no matched principle
Resolution of Conflict B
If(tic? tia) (tic=tjc) (pi=A ? pj=B)
Yes
No
Show result: no conflict C
Ask user for extra information of goal
If there are recommendations satisfied
Calculate the weight of the recommendation
Show results: recommendation by weight from big to small
Show results: no corresponding recommendation for this problem
Resolution of Conflict C
End

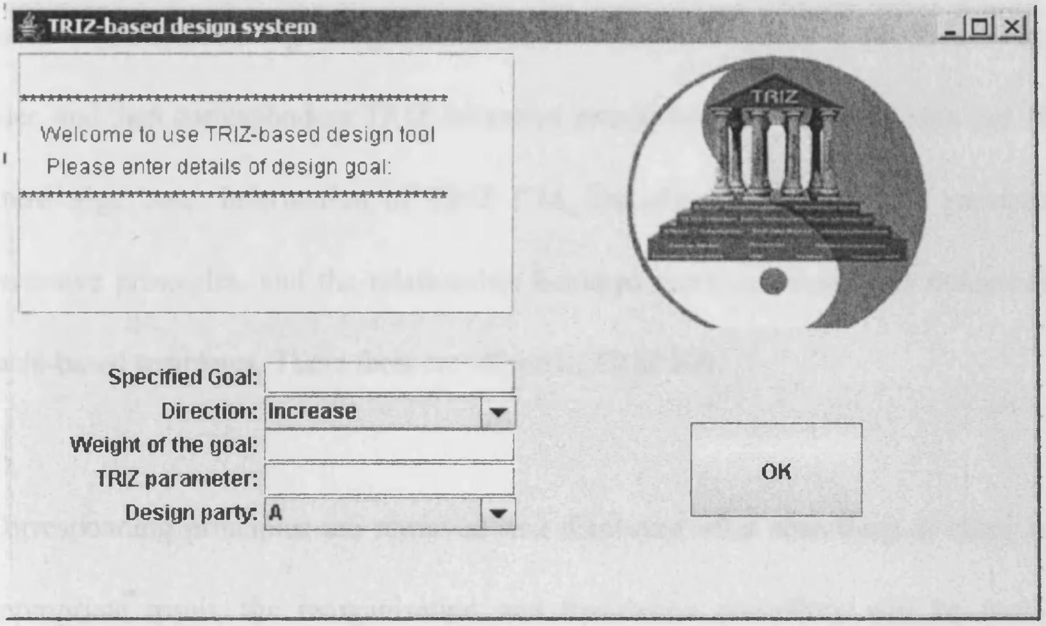
Figure 5.4 Flowchart of the system

The detection module starts by asking the user for goal information. The screenshot of user-input is shown in Figure 5.5. This action is activated by a routine rule, shown below.

```
(defrule ask-user-goal-information
   ?a <-(trigger-collaboration (id 1))
   =>
 (retract ?a)
 (goal-information)
 ((engine) waitForActivations)
      )
```

As core rules, three detection rules will be fired according to the information acquired by users. Thus, the type of conflict is detected. The following is the example of a detection rule for conflict A.

```
(defrule conflict-A-detection-rule
  (trigger-collaboration (id 2))
  (MAIN::Goal (weight ?w1) (T-parameter ?t1)(name ?n1))
  (MAIN::Goal (weight ?w2) (T-parameter ?t2)(name ?n2))
   (MAIN::final-contradiction-constraint-with-cell
   (first-objective ?t1)
   (second-objective ?t2))
   =>
   (if (>= (* ?w1 ?w2) 90)
      then
     (assert (conflict-information (catelgoue A)(first-t ?t1)(second-t ?t2)(first-
goal ?n1)(second-goal ?n2)))
     (assert (trigger-collaboration (id 3)))
     (assert (conflictA-and-B (t1 ?t1) (t2 ?t2)))))
```


TRIZ-based design system

***********************************************

Welcome to use TRIZ-based design tool

Please enter details of design goal:

***********************************************

Specified Goal:

Direction: Increase

Weight of the goal:

TRIZ parameter:

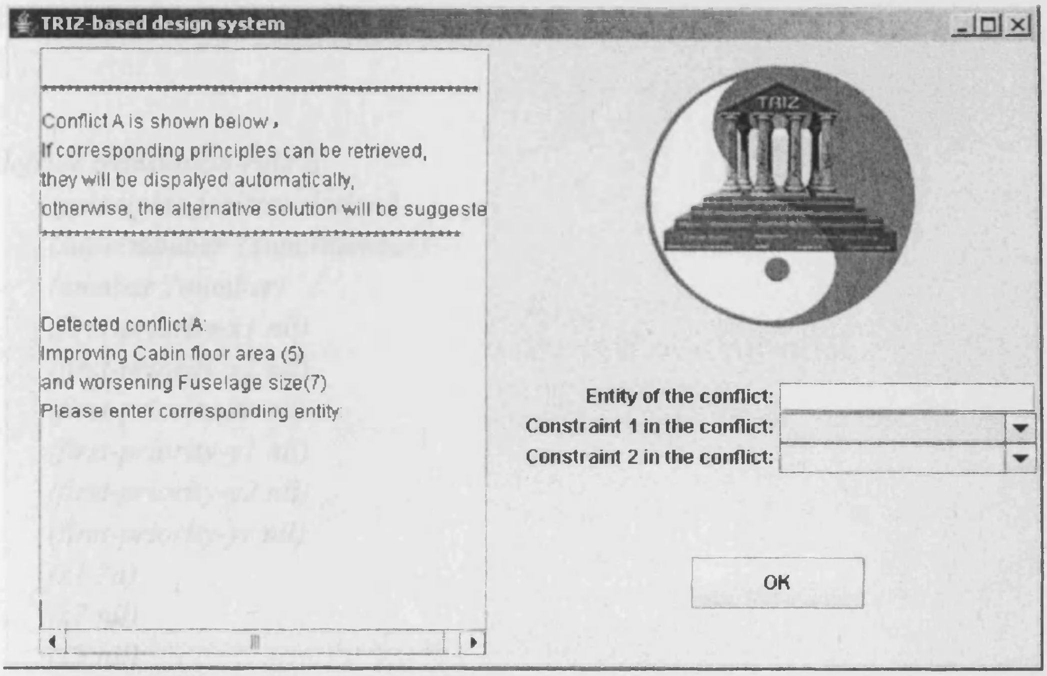
Design party: A

Figure 5.5 Screenshot of the detection module

### 5.4.4.2 Conflict A resolution module

The conflict A resolution module is initiated once conflict A has been detected in the detection module. Figure 5.6 shows the display on the computer screen when the module runs. At the beginning, additional information of entity is acquired from the user, and then correspondent TRIZ inventive principles are searched within the TRIZ knowledge base. Information of TRIZ CM, including row or column parameters, inventive principles, and the relationship between them are written as thousands of facts-based templates. These facts are stored in TRIZ KB.

Corresponding principles are retrieved and displayed after searching. If there is no appropriate result, the reorganisation and translation procedure will be initiated. Similarly, symbolic expression of inventive principles are written as facts, and stored in the KB. According to the approach presented in Chapter 4, the new solutions are generated by reorganizing BEC representation of inventive principles. Finally, the symbolic expression is translated into text.



TRIZ-based design system

*******************************************************

Conflict A is shown below ,
If corresponding principles can be retrieved,
they will be dispalyed automatically,
otherwise, the alternative solution will be suggeste

*******************************************************

Detected conflict A:
Improving Cabin floor area (5)
and worsening Fuselage size(7)
Please enter corresponding entity

Entity of the conflict:

Constraint 1 in the conflict:

Constraint 2 in the conflict:

Figure 5.6 Screenshot of the resolution module for type A conflicts

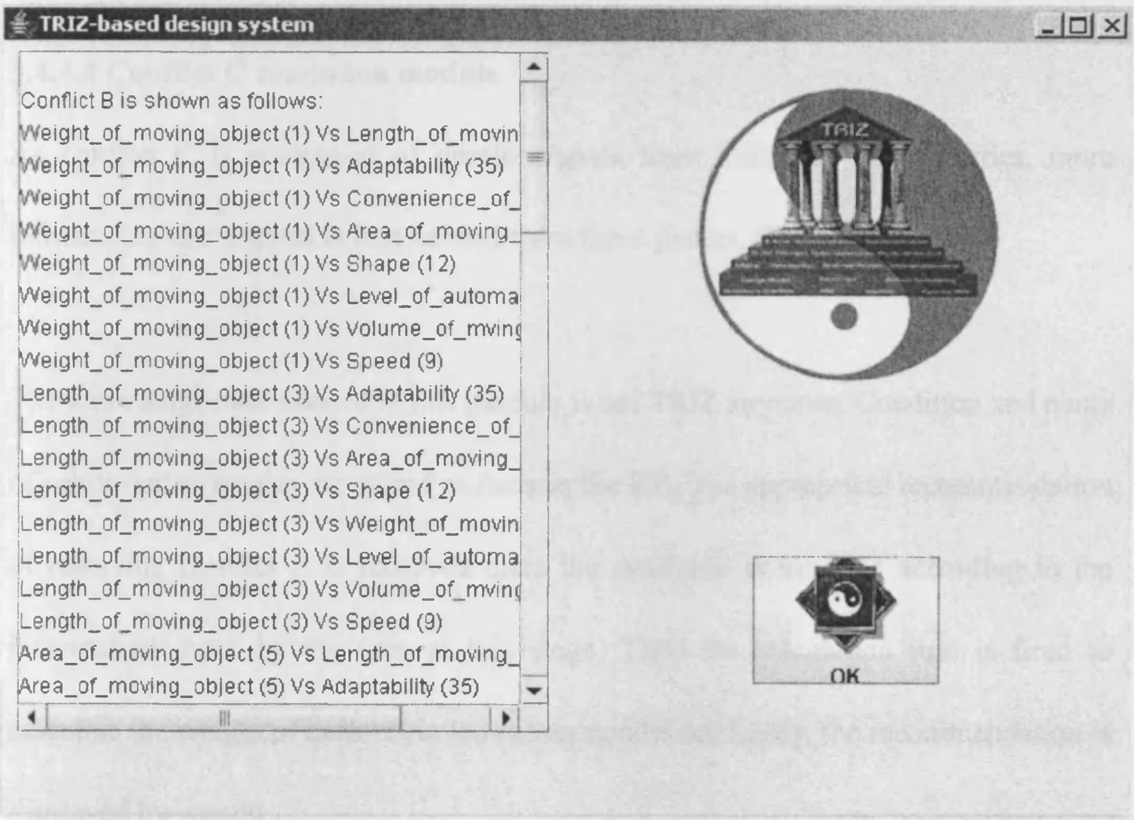
The core rules in this module include rule of retrieving principles, rule of organisation, and rule of translation. As described in chapter 3, there are 12 types of symbolic expression. The number of translation rules is therefore 12 as well, and one of them is shown below.

```
(defrule translation-rule-a
      (principles-for-translation2
      (supernumber ?supernumber)
      (number ?number)
      (first-priority-x1 nil)
      (first-priority-x2 nil)
      (first-priority-xr nil)
      (first-priority-y1 nil)
      (first-priority-y2 nil)
      (first-priority-yr nil)
      (x1 ?a)
      (x2 nil)
      (x3 nil)
      (xr1 nil)
      (xr2 nil)
      (xrf nil)
      (y1 ?b)
      (y2 nil)
      (y3 nil)
      (yr1 nil)
      (yr2 nil)
      (yrf nil)
      (so-symbol nil)
      (ex1 nil)
      (ex2 nil)
      )
   (x-axis (number ?a)
      (explanation1 ?c)
      (explanation2 ?d))
   (y-axis (number ?b)
      (name ?e))
   =>
(bind ?t (remove-nil ?c ?e ?d "" "" "" ""))
 (if (numberp ?supernumber)
   then
      (assert (translation
      (translation ?t)
      (number ?number)))
else
   (assert (translation
   (half-translation ?t)
   (supernumber ?supernumber)))
)
   )
```

### 5.4.4.3 Conflict B resolution module

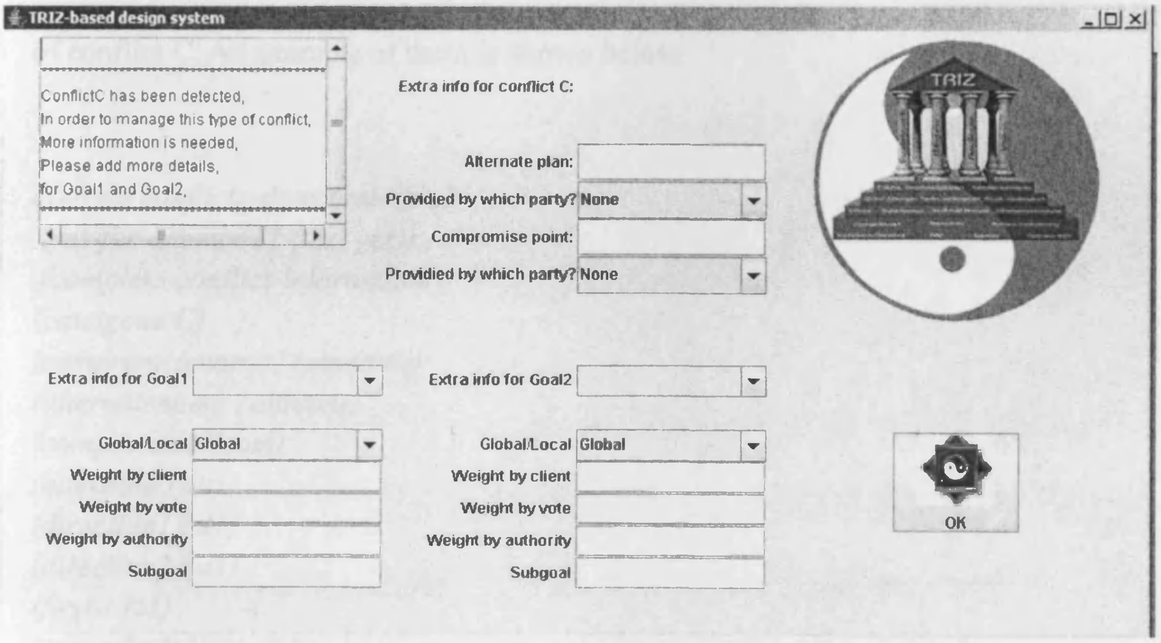
Conflict B resolution module is initiated once conflict B is detected. Figure 5.7 shows the screenshot of running this module. No additional user requirement is needed in this module. All the conflicting TRIZ parameters are displayed at the start of this module. The principles are then searched and retrieved within the same TRIZ KB in conflict A resolution module. The weight of each principle is then calculated and the principles are displayed by weight. Rules of retrieval and rules of calculation are the core rules in this module. An example of calculation rule is given below.

```
(defrule calculate-weight-and-frequency
   (weight-for-each-principle(principle ?p) (weight ?w))
   (frequency-of-principles (frequency ?n)(principles ?p))
    =>
      (bind ?nw (* ?w ?n))
      (assert (frequency-of-principles (weight ?nw)(principles ?p) (frequency ?n)))
      (bind ?s "Principles are shown by the importance from large to small")
   (?*qfield* setText ?s)
  )
```



Conflict B is shown as follows:
Weight_of_moving_object (1) Vs Length_of_movin
Weight_of_moving_object (1) Vs Adaptability (35)
Weight_of_moving_object (1) Vs Convenience_of_
Weight_of_moving_object (1) Vs Area_of_moving_
Weight_of_moving_object (1) Vs Shape (12)
Weight_of_moving_object (1) Vs Level_of_automa
Weight_of_moving_object (1) Vs Volume_of_mving
Weight_of_moving_object (1) Vs Speed (9)
Length_of_moving_object (3) Vs Adaptability (35)
Length_of_moving_object (3) Vs Convenience_of_
Length_of_moving_object (3) Vs Area_of_moving_
Length_of_moving_object (3) Vs Shape (12)
Length_of_moving_object (3) Vs Weight_of_movin
Length_of_moving_object (3) Vs Level_of_automa
Length_of_moving_object (3) Vs Volume_of_mving
Length_of_moving_object (3) Vs Speed (9)
Area_of_moving_object (5) Vs Length_of_moving_
Area_of_moving_object (5) Vs Adaptability (35)

Figure 5.7 Screenshot of the resolution module for type B conflicts



ConflictC has been detected,
In order to manage this type of conflict,
More information is needed,
Please add more details,
for Goal1 and Goal2,

Extra info for conflict C:

Alternate plan:
Providied by which party? None
Compromise point:
Providied by which party? None

Extra info for Goal1
Global/Local Global
Weight by client
Weight by vote
Weight by authority
Subgoal

Extra info for Goal2
Global/Local Global
Weight by client
Weight by vote
Weight by authority
Subgoal

Figure 5.8 Screenshot of the resolution module for type C conflicts

### 5.4.4.4 Conflict C resolution module

As conflict C is composed of opinions/goals from different design parties, more information about goals is first needed from these parties. (Figure 5.8)

The knowledge base needed in this module is not TRIZ anymore. Condition and name of resolution strategies are stored as facts in the KB. The appropriate recommendation of resolving conflict C is retrieved once the condition is satisfied according to the information input by the user at this stage. Then the calculation rule is fired to calculate the weight of each retrieved recommendation. Lastly, the recommendation is displayed by weight.

Calculation rules in conflict C resolution module are varied according to the category of conflict C. An example of them is shown below.

```
(defrule conflictc-detection-rule1
 (yes-for-acombo11 (text yes))
 (complete-conflict-information
(catelgoue C)
(compromisename ?comname)
(alternatename ?altname)
(compromise ?com)
(alternate ?alt)
(direction1 ?d1)
(direction2 ?d2)
(first-t ?t1)
(second-t ?t2)
(weight1 ?w1)
(weight2 ?w2)
(first-goal ?n1)
(second-goal ?n2)
(Local/global1 ?lg1)
(Local/global2 ?lg2)
(authorityweight1 ?aw1)
(authorityweight2 ?aw2)
(catelogue1 ?c1)
(catelogue2 ?c2)
(clientweight1 ?cw1)
(clientweight2 ?cw2)
(subgoal1 ?sg1)
(subgoal2 ?sg2)
(voteweight1 ?vw1)
(voteweight2 ?vw2))
     =>
(bind ?abandong1 (str-cat "Abandon Goal1: " ?n1))
(bind ?abandong2 (str-cat "Abandon Goal1: " ?n2))
(bind ?alt' (str-cat "Try alternate plan: " ?altname))

(bind ?sg1' (str-cat "Try alternate subgoal " ?sg1))
(bind ?sg2' (str-cat "Try alternate subgoal " ?sg2))

;;subgoal
(if (neq ?sg1 nil)
     then
     (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?sg1')(value
80)(party "Design parties"))))
(if (neq ?sg2 nil)
     then
     (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?sg2')(value
80)(party "Design parties"))))
     )
```

### 5.4.5 Example: Aircraft Fuselage

One of the principal purposes of air transport is to carry people comfortably and safely throughout their journey. The cabin interior will be carefully assessed by the customer. The actual cabin arrangement for a commercial aircraft is determined by marketing.

At the same time, the technical features such as size and weight of aircraft fuselage are often determined by regulations and other technical standards.

Therefore, design of passenger cabin layout and technical features of fuselage are usually responsible by different collaborative design parties. An example of the objective tree of this specified design is shown in Figure 5.9.

It can be seen that some functional objectives are in conflict with each other. A classical compromise is the one between the provision of sufficient space and cabin service to make the passengers feel comfortable during the flight, and the minimisation of the fuselage size to reduce structural weight and aerodynamic drag.

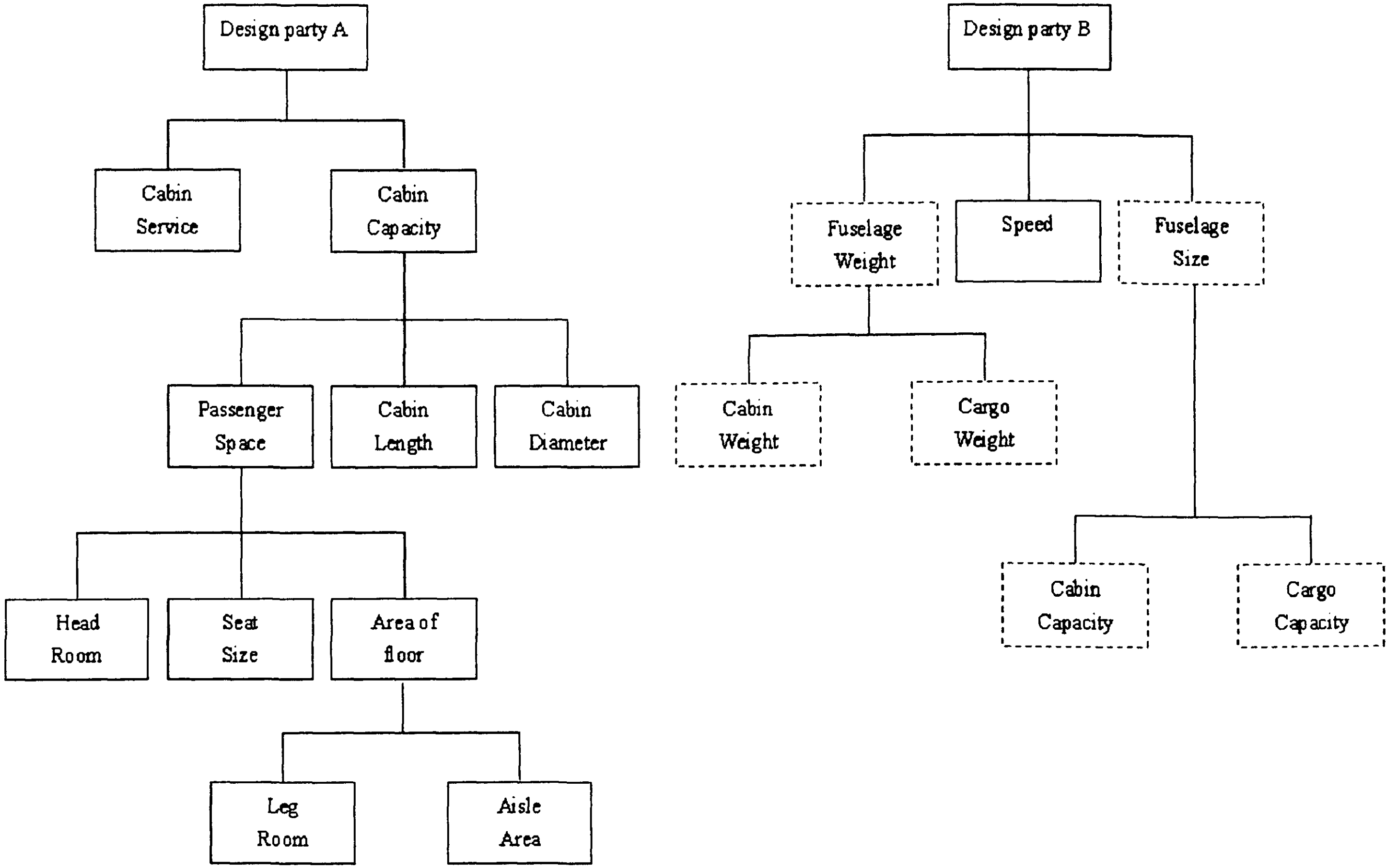


Figure 5.9 Figure 5.9 Objective trees for collaborative design of fuselage

In classical TRIZ, these contradictions are solved individually by retrieving inventive principles from the contradiction matrix. The limitations of applying classical TRIZ to collaborative conceptual design are as follows:

- The objectives of design are regarded as having the same priority. However, the importance of design objectives varies.
- The objectives of design are solved independently. They are not regarded as a system, which will lead to further contradictions.
- Certain conflicts in collaborative design cannot be retrieved from CM because of the limitations of the contradiction matrix itself.

All of these three limitations are avoided in the new system, the details of which will be described and demonstrated in the following sections.

The program starts by asking the user for general goal information. The user input is given in table 5.2 according to the objective trees and user requirements. The screenshot for this function is shown in figure 5.5.

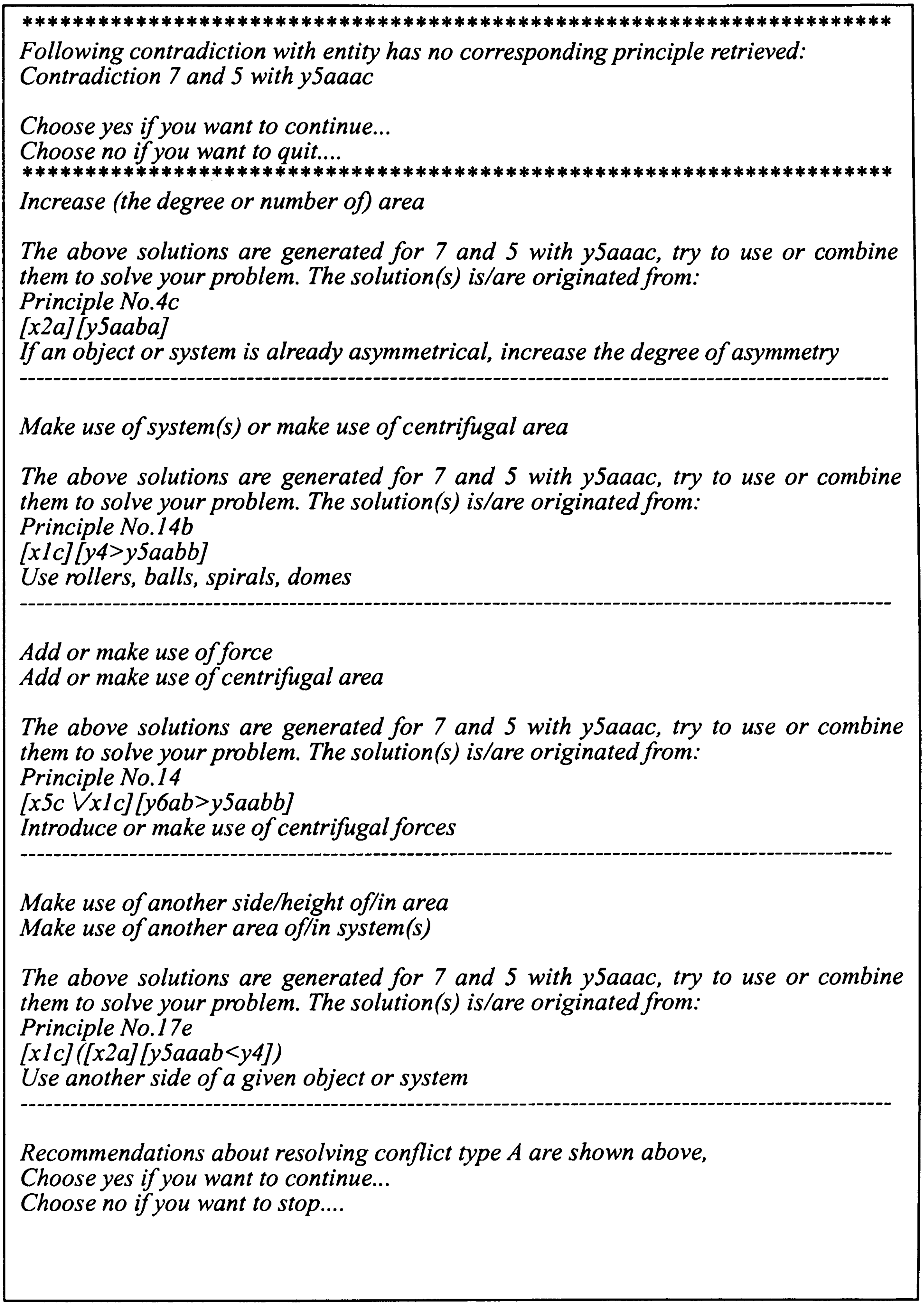
| **Goal name** | **Direction** | **Weight** | **TP** | **Party** |
|---|---|---|---|---|
| Cabin capacity | Increase | 10 | 7 | A |
| Cabin service | Increase | 8 | 35,33,38 | A |
| Cabin length | Increase | 7 | 3 | A |
| Cabin diameter | Increase | 8 | 5,12 | A |
| Area of floor | Increase | 10 | 5 | A |
| Fuselage weight | Decrease | 8 | 1 | B |
| Fuselage size | Decrease | 8 | 7 | B |
| Cabin capacity | Decrease | 8 | 7 | B |
| Cargo capacity | Decrease | 7 | 7 | B |
| Speed | Increase | 7 | 9 | B |

Table 5.2 Input data for goal information

One conflict of conflict type A has been detected first. A corresponding entity is then needed for further processing. (Figure 5.6) The case study in chapter 4 is implemented here. "Area" is then input as an entity for the same conflict in the case study. The result is shown in Figure 5.10.

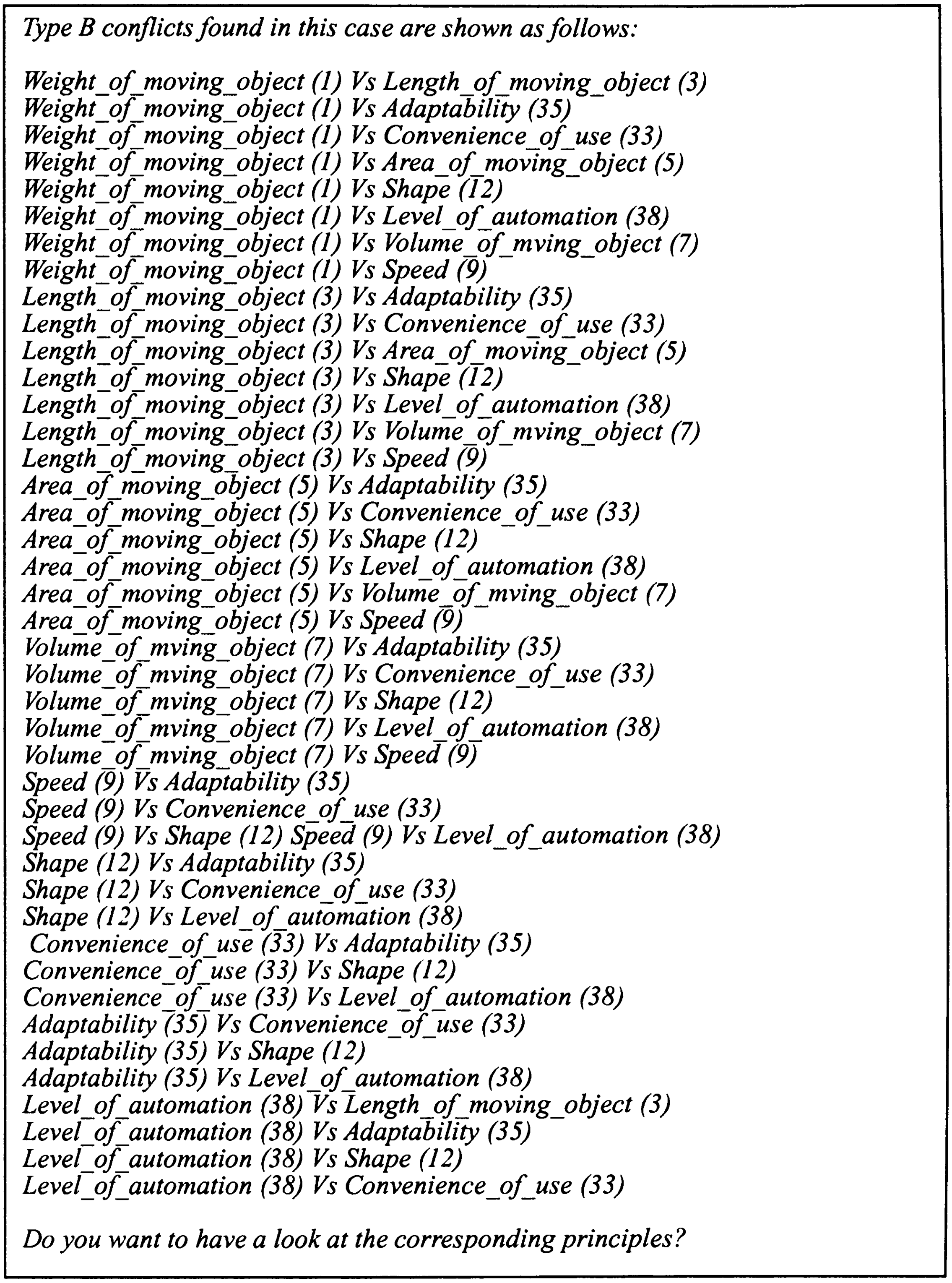
As there are no matched principles for this set of "constraint" and "entity", new solutions are generated by reorganizing classical inventive principles. These new solutions and their original principles are displayed as results at this stage. (Figure 5.10)

Compared with classical TRIZ, the result shown here provides more solutions and more specified principles for the user to choose. As conflict A is what users attach importance to, this result gives them more opportunity and wider space to investigate suitable solutions for conflicting problems.



```
*************************************************************************
Following contradiction with entity has no corresponding principle retrieved:
Contradiction 7 and 5 with y5aaac

Choose yes if you want to continue...
Choose no if you want to quit....
*************************************************************************
Increase (the degree or number of) area

The above solutions are generated for 7 and 5 with y5aaac, try to use or combine
them to solve your problem. The solution(s) is/are originated from:
Principle No.4c
[x2a][y5aaba]
If an object or system is already asymmetrical, increase the degree of asymmetry
------------------------------------------------------------------------------------------

Make use of system(s) or make use of centrifugal area

The above solutions are generated for 7 and 5 with y5aaac, try to use or combine
them to solve your problem. The solution(s) is/are originated from:
Principle No.14b
[x1c][y4>y5aabb]
Use rollers, balls, spirals, domes
------------------------------------------------------------------------------------------

Add or make use of force
Add or make use of centrifugal area

The above solutions are generated for 7 and 5 with y5aaac, try to use or combine
them to solve your problem. The solution(s) is/are originated from:
Principle No.14
[x5c Vx1c][y6ab>y5aabb]
Introduce or make use of centrifugal forces
------------------------------------------------------------------------------------------

Make use of another side/height of/in area
Make use of another area of/in system(s)

The above solutions are generated for 7 and 5 with y5aaac, try to use or combine
them to solve your problem. The solution(s) is/are originated from:
Principle No.17e
[x1c]([x2a][y5aaab<y4])
Use another side of a given object or system
------------------------------------------------------------------------------------------

Recommendations about resolving conflict type A are shown above,
Choose yes if you want to continue...
Choose no if you want to stop....
```

Figure 5.10 Result of conflict A resolution



```
Type B conflicts found in this case are shown as follows:

Weight_of_moving_object (1) Vs Length_of_moving_object (3)
Weight_of_moving_object (1) Vs Adaptability (35)
Weight_of_moving_object (1) Vs Convenience_of_use (33)
Weight_of_moving_object (1) Vs Area_of_moving_object (5)
Weight_of_moving_object (1) Vs Shape (12)
Weight_of_moving_object (1) Vs Level_of_automation (38)
Weight_of_moving_object (1) Vs Volume_of_mving_object (7)
Weight_of_moving_object (1) Vs Speed (9)
Length_of_moving_object (3) Vs Adaptability (35)
Length_of_moving_object (3) Vs Convenience_of_use (33)
Length_of_moving_object (3) Vs Area_of_moving_object (5)
Length_of_moving_object (3) Vs Shape (12)
Length_of_moving_object (3) Vs Level_of_automation (38)
Length_of_moving_object (3) Vs Volume_of_mving_object (7)
Length_of_moving_object (3) Vs Speed (9)
Area_of_moving_object (5) Vs Adaptability (35)
Area_of_moving_object (5) Vs Convenience_of_use (33)
Area_of_moving_object (5) Vs Shape (12)
Area_of_moving_object (5) Vs Level_of_automation (38)
Area_of_moving_object (5) Vs Volume_of_mving_object (7)
Area_of_moving_object (5) Vs Speed (9)
Volume_of_mving_object (7) Vs Adaptability (35)
Volume_of_mving_object (7) Vs Convenience_of_use (33)
Volume_of_mving_object (7) Vs Shape (12)
Volume_of_mving_object (7) Vs Level_of_automation (38)
Volume_of_mving_object (7) Vs Speed (9)
Speed (9) Vs Adaptability (35)
Speed (9) Vs Convenience_of_use (33)
Speed (9) Vs Shape (12) Speed (9) Vs Level_of_automation (38)
Shape (12) Vs Adaptability (35)
Shape (12) Vs Convenience_of_use (33)
Shape (12) Vs Level_of_automation (38)
 Convenience_of_use (33) Vs Adaptability (35)
Convenience_of_use (33) Vs Shape (12)
Convenience_of_use (33) Vs Level_of_automation (38)
Adaptability (35) Vs Convenience_of_use (33)
Adaptability (35) Vs Shape (12)
Adaptability (35) Vs Level_of_automation (38)
Level_of_automation (38) Vs Length_of_moving_object (3)
Level_of_automation (38) Vs Adaptability (35)
Level_of_automation (38) Vs Shape (12)
Level_of_automation (38) Vs Convenience_of_use (33)

Do you want to have a look at the corresponding principles?
```

Figure 5.11 List of retrieved conflict B

```
Principles are shown by the importance from large to small

Principle No. 29 appears 51 times.
The possiblity of solving problems by using this principle is 4080
Meaning: Pneumatics and hydraulics
Expressions:
[x7a][y6cb∨y6cc]
---------------------------------------------------------------------------------------

Principle No. 4 appears 51 times.
The possiblity of solving problems by using this principle is 4080
Meaning: Asymmetry
Expressions:
[x5c][y5aaba]
[x4c][y5aa] <- [y5aaba<y3]
[x2a][y5aaba]
---------------------------------------------------------------------------------------

Principle No. 35 appears 38 times.
The possiblity of solving problems by using this principle is 3040
Meaning: Parameter changes
Expressions:
[x4][y6c]
[x4][y8aa]
Repeated
[x4][y7aa]
[x4][y6aba]
---------------------------------------------------------------------------------------

Principle No. 17 appears 34 times.
The possiblity of solving problems by using this principle is 2176
Meaning: Another dimension
Expressions:
[x2a][y5aa]
repeated
[x2a][y5aaab]
[x1c][y6aaa<y4]
[x1c]([x2a][y5aaab<y4])
---------------------------------------------------------------------------------------

Principle No. 7 appears 29 times.
The possiblity of solving problems by using this principle is 2030
Meaning: Nested doll
Expressions:
[x5][y4>y5aba]
---------------------------------------------------------------------------------------

Principle No. 13 appears 24 times.
The possiblity of solving problems by using this principle is 1920
Meaning: The other way round
Expressions:
[x3][y1]
[x3c][y4>y6aa]
[x3][y5aaaa<y4]
```

Figure 5.12 Result of conflict B resolution

Resolution of all these conflicts will be a complex and time-consuming task. Moreover, new conflicts may be added (generated) during the resolution of current conflicts. In order to reduce the complexity of the task, the system will find the principles that solve as many conflicts as possible at the same time. Figure 5.11 and figure 5.12 show all the possible conflicts in this case and the results of desired principles arranged by weight and the number of conflicts that they can solve. It can be seen that principles No. 29, 4, and 15 appear more than 50 times, which should be considered as the key solutions to the problems.

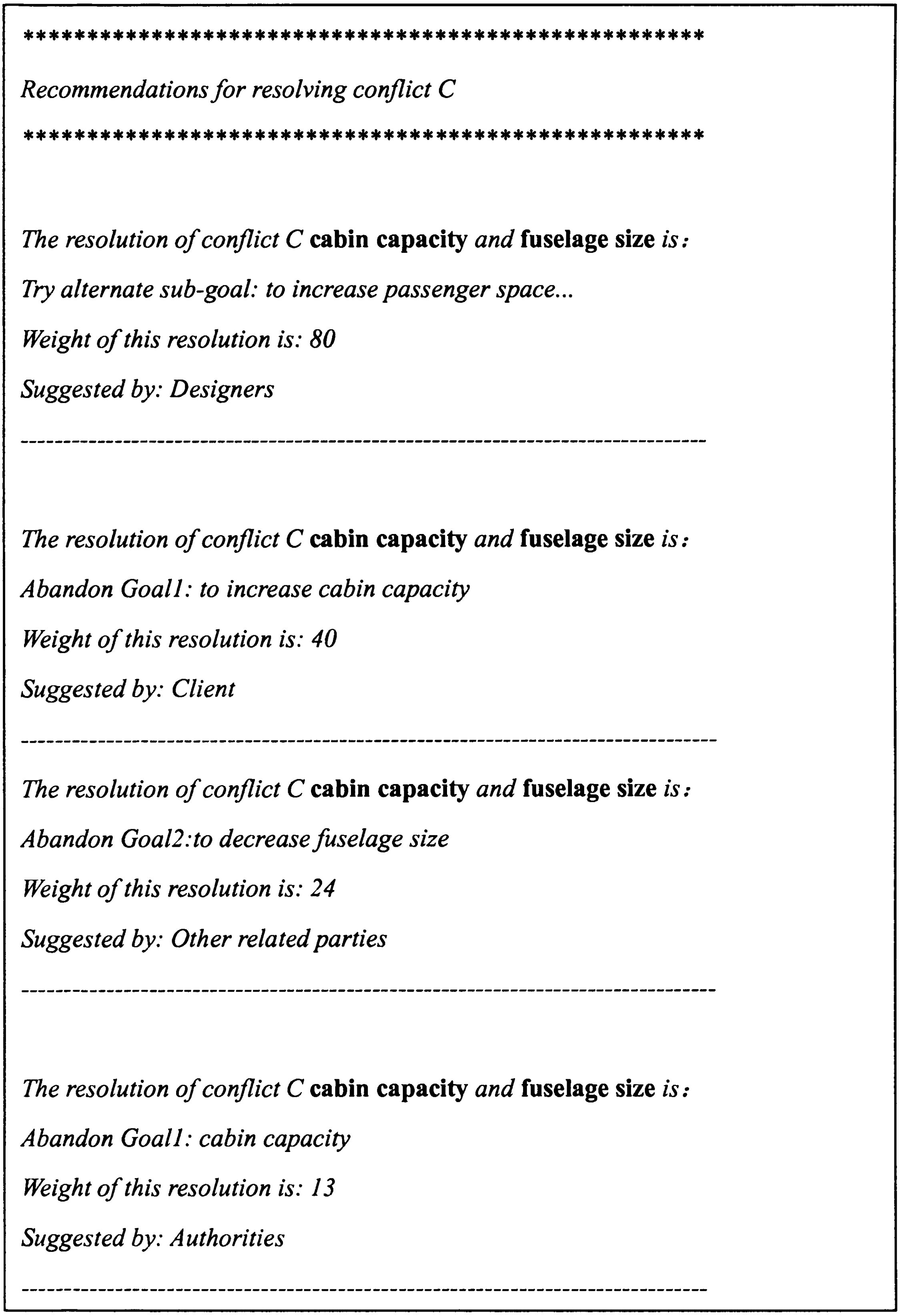
Three pairs of conflicts are identified as conflict type C in this case. Additional information on these goals is required including the weight of goal assigned by different parties, sub-goal, and alternative plan etc. They are shown in Table 5.3.

|  | AP |  | CP |  | G/L | CW | VW | AW | Sub-goal |
|---|---|---|---|---|---|---|---|---|---|
|  | N | Party | N | Party |  |  |  |  |  |
| Cabin capacity ( ↑ ) |  |  |  |  | L | 8 | 7 | 6 | Passenger space... |
| Fuselage size ( ↓ ) |  |  |  |  | G | 7 | 6 | 5 | Cabin capacity... |
| Cabin capacity( ↑ ) |  |  | 80 | A | L | 4 | 5 | 6 | Passenger space... |
| Cabin capacity( ↓ ) |  |  |  |  | L | 6 | 7 | 8 |  |
| Cabin capacity ( ↑ ) |  |  |  |  | L | 4 | 5 | 6 | Passenger space... |
| Cargo capacity ( ↑ ) |  |  |  |  | L | 5 | 4 | 3 |  |

Table 5.3 Input data for extra goal information

Figure 5.13 shows the recommendations for resolving conflicts in the design of an aircraft fuselage. For example, two conflicting goals “to increase cabin capacity” and “to decrease fuselage size” are identified as conflict type C which cannot be resolved in modules A and B. Four recommendations are presented automatically for the designer’s reference according to the information input by the different parties. The preferred solution is to try to satisfy the alternative sub-goal “to increase passenger space”. If this solution fails, the next preferred solutions are suggested. The ideal resolution strategy is the one to which both collaborative design parties have no objection . If agreement cannot be reached between them, the resolution suggested by the authorities will be used for resolving this difficult conflict.

In general, negotiation is a widely used approach for conflict resolution. Previous chapters have provided a no-compromise TRIZ-based technique for resolving conflicts and generating design concepts. However, in some circumstances, especially in a collaborative environment, conflicts cannot be resolved by this TRIZ-based approach. Three types of conflicts have been defined and corresponding resolution strategies have been suggested to overcome the limitations of the previous approach.



```
*****************************************************
Recommendations for resolving conflict C
*****************************************************


The resolution of conflict C cabin capacity and fuselage size is:
Try alternate sub-goal: to increase passenger space...
Weight of this resolution is: 80
Suggested by: Designers
--------------------------------------------------------------------------------


The resolution of conflict C cabin capacity and fuselage size is:
Abandon Goal1: to increase cabin capacity
Weight of this resolution is: 40
Suggested by: Client
--------------------------------------------------------------------------------
The resolution of conflict C cabin capacity and fuselage size is:
Abandon Goal2:to decrease fuselage size
Weight of this resolution is: 24
Suggested by: Other related parties
--------------------------------------------------------------------------------


The resolution of conflict C cabin capacity and fuselage size is:
Abandon Goal1: cabin capacity
Weight of this resolution is: 13
Suggested by: Authorities
--------------------------------------------------------------------------------
```

Figure 5.13 Result of conflict C resolution

## 5.5 Summary

This chapter defines and classifies three types of conflicts in conceptual design. Rules for detecting these conflicts are described. The resolution of each conflict is presented. Special attention is paid to the resolution of conflict C. The taxonomy of resolution strategies for conflict C and the conditions for using them is suggested. Moreover, the rules for calculating the weight of recommendation of resolution strategy are given. Lastly, the approach is implemented by using rule-based language JESS. The case study of fuselage design is applied to demonstrate the benefits of using this conflict resolution system. Also, the third hypothesis, "integration of TRIZ and negotiation-based methods can provide inventive conflict resolution strategies", has been proved by this case study.

# Chapter 6. Conclusion

## 6.1 Overview

This chapter presents the conclusions of this work, outlines the main contributions of the research and makes recommendations for further studies.

## 6.2 Conclusions

In Chapter 1, the objectives of this research were presented. These were:

- To evolve TRIZ inventive principles and apply them to supporting creative concept generation.
- To integrate negotiation-based approach and TRIZ to supporting conflict resolution.
- To provide computer support for this concept generation and conflict resolution at the stage of conceptual design, using KBS.

The hypotheses of this research were also presented in Chapter 1. These were as follows.

1. I-Ching can be a suitable tool for modifying TRIZ inventive principles.
2. Modified TRIZ-based design methodology can be more efficient than classical TRIZ.
3. Integration of TRIZ and negotiation-based methods can provide inventive conflict

resolution strategies.

The first hypothesis has been validated as nine new inventive principles have been generated based on I-Ching-inspired symbolic expression as explained in Chapter 3. The second and third hypotheses have been proved by case studies reported in Chapter 4 and Chapter 5.

Conceptual design is a very important and difficult task in an engineering product development cycle. It is the phase that requires expertise because the most important decisions taken at this stage are based on imprecise and incomplete knowledge of the design requirements and constraints. There are many domain dependent tools to support conceptual design in existence, while there are only a few that are actually domain independent. Moreover, conceptual design is a crucial stage when designing a new and innovative product, or when generating a completely new design for an existing product. Therefore, there is a demand for developing a domain independent approach to improve creativity in conceptual design.

This research has focused on the central activity of the conceptual design process, namely the generation of design concepts. Concept generation involves crucial activities that determine the final result of the developed product. It is also where designers require creativity to achieve design objectives. Here, TRIZ, an inventive problem-solving tool, has been applied to generate creative design concepts.

This research modifies TRIZ theory instead of applying original TRIZ to conceptual design. The inventive principles were extended by integrating other TRIZ tools and TRIZ-derived tools. These principles are also restructured by the inspiration of I-Ching. The new symbolic expression of inventive principles enables the generation of new and innovative solutions based on TRIZ.

This research has achieved not only the automatic retrieval of both existing and expanded principles as solutions according to the conflicting design requirements, but also the automatic generation of new solutions by reorganising the BEC representation of principles-based user requirements.

Negotiation is a widely used approach for conflict resolution. Conflict resolution is another focus area of this work. The research has provided a non-compromise technique for resolving conflicts based on the approach described above. However, the conflicts that occur in some circumstances cannot be resolved by this TRIZ-based approach. Another two types of conflict are defined and the corresponding resolution strategies are suggested to overcome the limitations of this TRIZ-based approach.

This research has presented software for automatically detecting and resolving conflicts according to the design requirements and the number of design parties. The recommendation is given as an output arranged by weight to help the designer improve creativity and efficiency for concept generation and conflict resolution in conceptual design.

## 6.3 Contributions

The research presented in this thesis has achieved the automatic generation of inventive design solutions and conflict resolution at the stage of conceptual design, including collaborative environment. The specific contributions are summarised below.

The research has contributed to the area of creative design. In particular, modification of a inventive problem solving tool, TRIZ. It addresses the reorganisation of TRIZ inventive principles based on the I-Ching inspired symbolic expression. As a result, redundant information has been removed and nine new principles have been produced. Also, the proposed symbolic expression of TRIZ principles enables the generation of creative design solutions according to design requirements.

Another contribution has been made to the area of conflict resolution. It addresses the detection and resolution of three types of conflicts in conceptual design. A non-compromise approach and a negotiation-based multi-stage approach have been integrated to improve the efficiency of resolving conflicts.

This work has also contributed to the area of intelligent computer-aided design. An artificial intelligence technology, KBS, has been selected to support conceptual design. Two domain independent knowledge bases have been built for the automatic retrieval of inventive design concepts and resolution strategies.

## 6.4 Future work

The following are possible topics for further study.

Expansion of the TRIZ knowledge base. Although the present TRIZ knowledge base has been modified by its integration with some TRIZ derived tools, more inventive design problems could be tackled by integrating the current KB with the other creative thinking tools described in this thesis.

Application of the TRIZ-based KBS into other areas. The system developed in this work can be modified and applied in the areas of social science or business, where problems can also be solved by TRIZ.

Integration of various representation schemes. Feature-based and structure-based models can be integrated with the function-based model in the present system to represent more aspects of design knowledge.

# Appendix A. TRIZ 40 inventive principles

## Principle 1. Segmentation

A. *Divide a system into separate parts or sections.*
B. *Make a system easy to put together or take apart*
C. *Increase the amount of segmentation*

## Principle 2. Taking Out

A. *Where a system provides several functions of which one or more is not required (and may be harmful) at certain conditions, design the system so they are or can be taken out.*

## Principle 3. Local Quality

A. *Where an object or a system is uniform or homogenous, make it non-uniform.*
B. *Change things around the system from uniform to non-uniform.*
C. *Enable each part of a system to function in locally optimized conditions.*
D. *Enable each part of a system or object to carry out different useful functions.*

## Principle 4. Asymmetry

A. *Where an object or system is symmetrical or contains lines of symmetry, introduce asymmetries.*
B. *Change the shape of an object or system to suit external asymmetries.*
C. *If an object or system is already asymmetrical, increase the degree of asymmetry.*

## Principle 5. Merging

A *Physically join or merge identical or related objects, operations or functions.*
B *Join or merge objects, operations or functions so that they act together in time.*

## Principle 6. Universality

A. *Make an object or system able to perform multiple functions, eliminating the need for other systems.*

## Principle 7. "Nested Doll"

*A* *Put one object inside another.*
*B* *Put several objects or systems inside others.*
*C* *Allow one object or system to pass through an appropriate hole in another.*

## Principle 8. Anti-Weight

*A.* *Where the weight of an object or system causes problems, combine it with something that provides lift or use aerodynamic, hydrodynamic, buoyancy or other forces to provide lift.*

## Principle 9. Preliminary Anti-Action

*A.* *Where an action contains both harmful and useful effects, precede the action with opposite or anti-actions to reduce or eliminate the harmful effects.*
B. *Introduce stresses in an object to oppose known harmful working stresses later on.*

## Principle 10. Preliminary Action

*A.* *Introduce a useful action into an object or system (fully or partially) before it is needed.*
B. *Pre-arrange objects or systems such that they can come into action at the most convenient time and place.*

## Principle 11. Beforehand Cushioning

A. *Introduce emergency backups to compensate for the potentially low reliability of an object (belt and suspenders).*

## Principle 12. Equipotentiality

A. *If an object or system requires or is exposed to tension or compression forces are eliminated or are balanced by the surrounding environment.*

## Principle 13. "The other way around"

A. *Use an opposite action(s) used to solve the problem.*
B. *Make moveable objects fixed, and the fixed objects movable.*
C. *Turn the object, system or process upside-down.*

## Principle 14. Spheroidality-Curvature

A. *Turn straight edges or flat surfaces into curves.*
B. *Use rollers, balls, spirals, domes.*
C. *Switch between linear and rotary motion.*
D. *Introduce or make use of centrifugal forces.*

## Principle 15. Dynamics

A. *Allow a system or object to change to achieve optimal operation under different conditions.*
B. *Split an object or system into parts capable of moving relative to each other.*
C. *If an object or system is rigid or inflexible, make it movable or adaptable.*
D. *Increase the amount of free motion.*

## Principle 16. Partial or Excessive Actions

A. *If exactly the right amount of an action is hard to achieve, use slightly less or slightly more of the action to reduce or eliminate the problem.*

## Principle 17. Another Dimension

A. *If an object contains or moves in a straight line, consider use of dimensions or movement outside the line.*
B. *If an object contains or moves in a plane, consider use of dimensions or movement outside the current plane.*
C. *Use a stacking arrangement of objects instead of a single level arrangement.*
D. *Re-orient the object or system, lay it on its side.*
E. *Use another side of a given object or system.*

## Principle 18. Mechanical Vibration

A. *Cause an object to oscillate or vibrate.*

*B.* *Increase the vibration frequency (ultrasonic).*  
*C.* *Make use of an object or systems resonant frequency.*  
*D.* *Use piezoelectric vibrators.*  
E. *Use combined field oscillations.*

## Principle 19. Periodic Action

*A.* *Replace continuous actions with periodic or pulsating actions.*  
*B.* *If an action is already periodic, change the periodic magnitude or frequency to suit external requirements*  
C. *Use gaps between actions to perform different useful actions.*

## Principle 20. Continuity of Useful Action

*A.* *Make all parts of an object or system work at full load or optimum efficiency all the time.*  
B. *Eliminate all idle or non-productive actions or work.*

## Principle 21. Skipping

*A* *Conduct an action a very high speed to eliminate harmful side effects.*

## Principle 22. "Blessing in Disguise"

*A.* *Transform harmful objects or actions (particularly, the environment or surroundings) so that they deliver a positive effect.*  
B. *Add a 2nd harmful object or action to neutralized or eliminate the effect of an existing harmful object or action; Increase a harmful factor to such a level that it no longer causes harm.*

## Principle 23. Feedback

*A.* *Introduce feedback to improve a process or action.*  
B. *Make current feedback adaptable to variations in operating requirements or conditions.*

## Principle 24. "Intermediary"

A. *Introduce an intermediary between two objects, systems or actions.*

B. *Introduce a temporary intermediately which disappears or can be easily removed after it has completed its function.*

## Principle 25. Self-Service

A. *Enable an object or system to perform functions by itself or organize itself.*

B. *Make use of waste resources energy or substances.*

## Principle 26. Copying

A. *Use simple and inexpensive copies in place of expensive, possibly vulnerable objects or systems.*

B. *Replace an object or action with an optical copy.*

C. *Use infrared or ultraviolet wavelengths with optical copies.*

## 27. Cheap Short-living Objects

A. *Replace an expensive object or system with a multitude of inexpensive, short-life objects.*

## 28. Mechanics Substitution / Another Sense

A. *Replace an existing means with a means of making use of another sense (optical, acoustic, taste, touch or smell).*

B. *Introduce electric, magnetic or electromagnetic fields to interact with an object or system.*

C. *Change from static to movable, fixed to variable, and/or from unstructured to structured fields.*

D. *Use fields in conjunction with field-activated objects or systems.*

## Principle 29. Pneumatics and Hydraulics

A *Use gases and liquids instead of solid parts or systems.*

## Principle 30. Flexible Shells and Thin Films

A. *Incorporate flexible shells and thin films instead of solid structures.*

B. *Isolate an object or system from a potentially harmful environment using flexible shells and thin films.*

## Principle 31. Porous Materials

A. *Make and object porous or add porous elements.*
B. *Add something useful into the pores.*

## Principle 32. Colour Changes

A. *Change the colour or an object or its surroundings.*
B. *Change the transparency of an object or its surroundings.*
C. *In order to change the visibility or things, use colour additives or luminescent elements.*
D. *Change the emissivity properties of an object subject to radiant heating.*

## Principle 33. Homogeneity

A. *Make interaction objects from the same material (or material with similar properties).*

## Principle 34. Discarding and Recovering

A. *Make elements of an object or system that have fulfilled their functions disappear (dissolving, etc.) or appear to disappear.*
B. *Restore consumable or degradable parts of an object or system during operation.*

## Principle 35. Parameter Changes

A. *Change an objects physical state.*
B. *Change concentration or consistency.*
C. *Change the degree of flexibility.*
D. *Change the temperature.*
E. *Change the pressure.*
F. *Change other parameters.*

## Principle 36. Phase Transitions

A. *Make use of phenomena taking place during phase transitions (volume changes,* *heat loss, etc.)*

## Principle 37. Thermal Expansion

*A.* *Use thermal expansion or contraction of materials to achieve a useful result.*
B. *Use multiple materials with different thermal expansion rates to achieve different useful effects.*

## Principle 38. Strong Oxidants

A. Replace atmospheric air with oxygen-enriched air.
B. Use pure oxygen.
C. Use ionizing radiation.
D. Use ionized oxygen.
E. Use ozone.

## Principle 39. Inert Atmosphere

*A.* *Replace a normal environment with an inert one.*
B. *Add neutral parts, or inert elements to an object or system.*

## Principle 40. Composite Material

*A.* *Change from uniform to composite (multiple) materials where each material is optimized to a particular functional requirement.*

# Appendix B. Code for detection module

```
(defmodule ask-goal)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;ask user if need to add more goals
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(defrule more-goal-or-not
      (Goal (T-parameter ?t&: (or (<= (integer ?t) 39)(>= (integer ?t) 1) ))
            (direction ?d&: (neq ?d nil))
            (name ?n&: (neq ?n nil))
            (party ?p&: (neq ?p nil))
            (weight ?w&: (or (<= (integer ?w) 10)(>= (integer ?w) 1) )))
      =>
      ;;(printout t "test")
  ;;(bind ?scroll (new JScrollPane ?*qfield*))
  ;;((?*frame-c* getContentPane) add ?scroll)
;;((?*frame-c* getContentPane) add ?*qfield*)
;;(?*qfield* setText "Please wait...")
      (?*apanel* removeAll)
  ;;      (?*qfield* setText "If you want to add more goals?")
            (?*apanel* add ?*scroll*)
      (?*apanel* add ?*alabel8*)
      (?*apanel* add ?*acombo8*)
      (?*apanel* add ?*acombo-ok8*)
    ;; ((?*frame-c* getContentPane) add ?*apanel* (get-member BorderLayout SOUTH))
    ((?*frame-c* getContentPane) add ?*apanel*)
       (?*frame-c* validate)
       (?*frame-c* repaint)
       ((engine) waitForActivations)
     )
(defrule add-more-goal
      ?a <-(user-input-collaboration (text yes))
      =>
      (retract ?a)
    (assert (trigger-collaboration (id 1))))

(defrule no-more-goal
      ?a <-(user-input-collaboration (text no))
      =>
      (retract ?a)
    (assert (trigger-collaboration (id 2))))
```

```
 ;; (facts)
        ((engine) waitForActivations)
     )

(defrule assert-entity-constraint-input
     ?user      <-(user-input-collaboration      (entity      ?text1)(constraint1      ?text2)
(constraint2 ?text3))
=>
    (retract ?user)
    (assert  (user-input-eandc (entity ?text1)(constraint1 ?text2)(constraint2 ?text3)) )
       (assert (trigger-collaboration (id 4)))
     ;;(bind ?s "The following principles are retrieved:
     ;;")
    ;;(?*qfield* setText ?s)
       )


(defmodule conflict-detectionA2-3)

(defrule retrieve-principle-in-eandc
    (trigger-collaboration (id 4))
    (user-input-eandc  (entity  ?e&:  (neq  ?e  nil))(constraint1  ?c1&:  (neq  ?c1  nil))
(constraint2 ?c2&: (neq ?c2 nil)))
(or                (principles-for-translation                (first-priority-y1               ?e)
(number ?n1)(expression ?ex1)(text ?t1)(supernumber ?supern1))
    (principles-for-translation                      (first-priority-y2                  ?e)
(number ?n2)(expression ?ex2)(text ?t2)(supernumber ?supern2))
     (principles-for-translation                     (y1                         ?e)
(number ?n3)(expression ?ex3)(text ?t3)(supernumber ?supern3))
     (principles-for-translation                     (y2                         ?e)
(number ?n4)(expression ?ex4)(text ?t4)(supernumber ?supern4))
     (principles-for-translation                     (y3                         ?e)
(number ?n5)(expression ?ex5)(text ?t5)(supernumber ?supern5))
     )

  (final-contradiction-constraint-with-cell (first-objective ?c1) (second-objective ?c2) (cell
$?cell))
      =>
    ;; (printout t "test")
;; (assert (trigger-collaboration (id 5)))
     (assert (temp-user-input-eandc (entity ?e)(constraint1 ?c1)(constraint2 ?c2)))

(print-entity-based-principle ?c1 ?c2 ?supern1 ?supern2 ?supern3 ?supern4 ?supern5 ?cell ?
n1 ?ex1 ?t1 ?n2 ?ex2 ?t2 ?n3 ?ex3 ?t3 ?n4 ?ex4 ?t4 ?n5 ?ex5 ?t5)
)

(defmodule conflict-detectionA2-3-1)

(defrule delete-used-eandc
     (temp-user-input-eandc (entity ?e)(constraint1 ?c1)(constraint2 ?c2))
     ?eandc <-(user-input-eandc (entity ?e&: (neq ?e nil))(constraint1 ?c1&: (neq ?c1 nil))
(constraint2 ?c2&: (neq ?c2 nil)))
     =>
     (retract ?eandc))


(defmodule conflict-detectionA3)
```

```
(defrule no-entity-retrived-before1
              (trigger-collaboration (id 4))
    ;; (not (trigger-collaboration (id 5)))
      =>
     (     ?*qfield* append "

*******************************************************************************
***
Following contradiction with entity has no corresponding principles retrieved:
          ")
     (assert (trigger-collaboration (id 13)))
     )

(defrule no-entity-retrieved-before

             (user-input-eandc (entity ?e&: (neq ?e nil))(constraint1 ?c1&: (neq ?c1 nil))
(constraint2 ?c2&: (neq ?c2 nil)))
         (trigger-collaboration (id 13))
      =>
         (bind ?a (str-cat "
contradiction " ?c1 " and " ?c2 " with " ?e ?*crlf*)
  )
    (?*qfield* append ?a)
  (assert (trigger-collaboration (id 14)))

     )

(defmodule conflict-detectionA3-4)

(defrule no-entity-based-principle-retrieved
     ;; (user-input-eandc (entity ?e&: (neq ?e nil))(constraint1 ?c1&: (neq ?c1 nil))
(constraint2 ?c2&: (neq ?c2 nil)))
       (trigger-collaboration (id 14))
           =>
     ;;(printout t "test1")
     (ask-user-if-generate-new-principles)
        ((engine) waitForActivations)
   ;;  (bind ?a (str-cat "The new solutions are generated " "for " ?c1 " and "?c2 " with "?e "
as follows, try to combine them to solve your problem:

;;"))
   ;; (?*qfield* append ?a)
     )

(defrule replace-and-search
       (once-yes-or-no (text yes))
     (user-input-eandc (entity ?e&: (neq ?e nil))(constraint1 ?c1&: (neq ?c1 nil))
(constraint2 ?c2&: (neq ?c2 nil)))
     (final-contradiction-constraint-with-cell (first-objective ?c1) (second-objective ?c2) (cell
$?cell))
     ;;find different level of corresponding entity
     (y-axis (number ?e)(supernumber ?supern))
     (y-axis (number ?m)(supernumber ?supern))
     (y-axis (number ?supern) (supernumber ?supersupern))
     (y-axis (number ?superm) (supernumber ?supersupern))
     (y-axis (number ?subn) (supernumber  ?superm))
      ;;replace with entity at same level
   (or
        (or (principles-for-translation (first-priority-y1 ?m)
(number ?n11)(expression ?ex11)(text ?t11)(supernumber ?supern11))
     (principles-for-translation (first-priority-y2 ?m)
(number ?n12)(expression ?ex12)(text ?t12)(supernumber ?supern12))
     (principles-for-translation (y1 ?m)
(number ?n13)(expression ?ex13)(text ?t13)(supernumber ?supern13))
     (principles-for-translation (y2 ?m)
(number ?n14)(expression ?ex14)(text ?t14)(supernumber ?supern14))
     (principles-for-translation (y3 ?m)
(number ?n15)(expression ?ex15)(text ?t15)(supernumber ?supern15))
     )
     ;;replace with entity at super level
          (or (principles-for-translation (first-priority-y1 ?supern&: (neq ?supern y0))
(number ?n21)(expression ?ex21)(text ?t21)(supernumber ?supern21))
    (principles-for-translation (first-priority-y2 ?supern&: (neq ?supern y0))
(number ?n22)(expression ?ex22)(text ?t22)(supernumber ?supern22))
     (principles-for-translation (y1 ?supern&: (neq ?supern y0))
(number ?n23)(expression ?ex23)(text ?t23)(supernumber ?supern23))
     (principles-for-translation (y2 ?supern&: (neq ?supern y0))
(number ?n24)(expression ?ex24)(text ?t24)(supernumber ?supern24))
     (principles-for-translation (y3 ?supern&: (neq ?supern y0))
(number ?n25)(expression ?ex25)(text ?t25)(supernumber ?supern25))
     )
     ;;replace with entity at super level's neighbour
        (or (principles-for-translation (first-priority-y1 ?superm)
(number ?n31)(expression ?ex31)(text ?t31)(supernumber ?supern31))
    (principles-for-translation (first-priority-y2 ?superm)
(number ?n32)(expression ?ex32)(text ?t32)(supernumber ?supern32))
     (principles-for-translation (y1 ?superm)
(number ?n33)(expression ?ex33)(text ?t33)(supernumber ?supern33))
     (principles-for-translation (y2 ?superm)
(number ?n34)(expression ?ex34)(text ?t34)(supernumber ?supern34))
     (principles-for-translation (y3 ?superm)
(number ?n35)(expression ?ex35)(text ?t35)(supernumber ?supern35))
     )
     ;;replace with entity at super level's sublevel
          (or (principles-for-translation (first-priority-y1 ?subn)
(number ?n41)(expression ?ex41)(text ?t41)(supernumber ?supern41))
    (principles-for-translation (first-priority-y2 ?subn)
(number ?n42)(expression ?ex42)(text ?t42)(supernumber ?supern42))
     (principles-for-translation (y1 ?subn)
(number ?n43)(expression ?ex43)(text ?t43)(supernumber ?supern43))
     (principles-for-translation (y2 ?subn)
(number ?n44)(expression ?ex44)(text ?t44)(supernumber ?supern44))
     (principles-for-translation (y3 ?subn)
(number ?n45)(expression ?ex45)(text ?t45)(supernumber ?supern45))
     ))
        =>
     ;; (printout t "testtest2")
     ;; (retract ?a)

(print-entity-based-principle2 ?e ?c1 ?c2 ?supern11 ?supern12 ?supern13 ?supern14 ?super
n15 ?cell ?n11 ?ex11 ?t11 ?n12 ?ex12 ?t12 ?n13 ?ex13 ?t13 ?n14 ?ex14 ?t14 ?n15 ?ex15 ?t1
5)

(print-entity-based-principle2 ?e ?c1 ?c2 ?supern21 ?supern22 ?supern23 ?supern24 ?super
n25 ?cell ?n21 ?ex21 ?t21 ?n22 ?ex22 ?t22 ?n23 ?ex23 ?t23 ?n24 ?ex24 ?t24 ?n25 ?ex25 ?t2
5)
```

```
(print-entity-based-principle2 ?e ?c1 ?c2 ?supern31 ?supern32 ?supern33 ?supern34 ?super
n35 ?cell ?n31 ?ex31 ?t31 ?n32 ?ex32 ?t32 ?n33 ?ex33 ?t33 ?n34 ?ex34 ?t34 ?n35 ?ex35 ?t3
5)

(print-entity-based-principle2 ?e ?c1 ?c2 ?supern41 ?supern42 ?supern43 ?supern44 ?super
n45 ?cell ?n41 ?ex41 ?t41 ?n42 ?ex42 ?t42 ?n43 ?ex43 ?t43 ?n44 ?ex44 ?t44 ?n45 ?ex45 ?t4
5)

          ;;(printout t "ceshichenggong")
          )

(defmodule conflict-detectionA4)

(defrule print-replaced-and-searched-result
   ;; (user-input-eandc (entity ?e&: (neq ?e nil))(constraint1 ?c1&: (neq ?c1 nil))
(constraint2 ?c2&: (neq ?c2 nil)))
      (templist-in-eandc2 (entity ?e)(constraint1 ?c1)(constraint2 ?c2)(expression ?expression)
(number   ?number) (text ?text))
      =>
   ;; (facts)
      (bind ?s (str-cat "
The above solutions are generated " " for " ?c1 " and "?c2 " with "?e " as follows, try to
combine them to solve your problem: "
"These solutions are originated from:
Principle   No."   ?number   ?*crlf*   ?expression   ?*crlf*   ?text   ?*crlf*
"----------------------------------------" ?*crlf*)
          )
      (?*qfield* append ?s)
   ;;(printout t "test")

      )

(defrule print-new-solution-with-original-entity
      (user-input-eandc (entity ?e))
      (templist-in-eandc2 (expression ?expression) (number   ?number) (text ?text))
      (principles-for-translation
(supernumber ?supernumber)(number ?number)(x1 ?x1)(x2 ?x2)(x3 ?x3)(xr1 ?xr1)
(xr2 ?xr2)(y1 ?y1) (y2 ?y2) (y3 ?y3)(yr1 ?yr1)(yr2 ?yr2) (first-priority-y1 ?y4)
(first-priority-y2 ?y5)(first-priority-yr ?fpyr)(first-priority-x1 ?x4) (first-priority-x2 ?x5)
(first-priority-xr ?fpxr))
      =>
      (if (neq ?y1 nil)
          then
          (assert                                   (principles-for-translation2
(number ?number)(supernumber ?supernumber) (y1 ?e)(x1 ?x1)(x2 ?x2)(x3 ?x3)(xr1 ?xr1)
(xr2 ?xr2)(y2 ?y2) (y3 ?y3)(yr1 ?yr1)(yr2 ?yr2) (first-priority-y1 ?y4)
(first-priority-y2 ?y5)(first-priority-yr ?fpyr)(first-priority-x1 ?x4) (first-priority-x2 ?x5)
(first-priority-xr ?fpxr))
                  ))
          (if (neq ?y2 nil)
          then
          (assert       (principles-for-translation2       (number             ?number)
(supernumber ?supernumber)(y2 ?e)(x1 ?x1)(x2 ?x2)(x3 ?x3)(xr1 ?xr1) (xr2 ?xr2)(y1 ?y1)
(y3        ?y3)(yr1         ?yr1)(yr2         ?yr2)         (first-priority-y1         ?y4)
(first-priority-y2 ?y5)(first-priority-yr ?fpyr)(first-priority-x1 ?x4) (first-priority-x2 ?x5)
(first-priority-xr ?fpxr))
                  ))
```

```
        (if (neq ?y3 nil)
        then
        (assert (principles-for-translation2 (number ?number)
(supernumber ?supernumber)(y3 ?e)(x1 ?x1)(x2 ?x2)(x3 ?x3)(xr1 ?xr1) (xr2 ?xr2)(y2 ?y2)
(y1 ?y1)(yr1 ?yr1)(yr2 ?yr2) (first-priority-y1 ?y4)
(first-priority-y2 ?y5)(first-priority-yr ?fpyr)(first-priority-x1 ?x4) (first-priority-x2 ?x5)
(first-priority-xr ?fpxr))
                ))
            (if (neq ?y4 nil)
        then
        (assert (principles-for-translation2 (number ?number)
(supernumber ?supernumber)(first-priority-y1 ?e)(x1 ?x1)(x2 ?x2)(x3 ?x3)(xr1 ?xr1)
(xr2 ?xr2)(y1 ?y1)(y2 ?y2) (y3 ?y3)(yr1 ?yr1)(yr2 ?yr2)
(first-priority-y2 ?y5)(first-priority-yr ?fpyr)(first-priority-x1 ?x4) (first-priority-x2 ?x5)
(first-priority-xr ?fpxr))
                ))
                (if (neq ?y5 nil)
        then
        (assert (principles-for-translation2 (number ?number)
(supernumber ?supernumber)(first-priority-y2 ?e)(x1 ?x1)(x2 ?x2)(x3 ?x3)(xr1 ?xr1)
(xr2 ?xr2)(y1 ?y1)(y2 ?y2) (y3 ?y3)(yr1 ?yr1)(yr2 ?yr2)
(first-priority-y1 ?y4)(first-priority-yr ?fpyr)(first-priority-x1 ?x4) (first-priority-x2 ?x5)
(first-priority-xr ?fpxr))
                ))
)


(defrule translation-rule-a
    (principles-for-translation2
        (supernumber ?supernumber)
        (number ?number)
        (first-priority-x1 nil)
        (first-priority-x2 nil)
        (first-priority-xr nil)
        (first-priority-y1 nil)
        (first-priority-y2 nil)
        (first-priority-yr nil)
        (x1 ?a)
        (x2 nil)
        (x3 nil)
        (xr1 nil)
        (xr2 nil)
        (xrf nil)
        (y1 ?b)
        (y2 nil)
        (y3 nil)
        (yr1 nil)
        (yr2 nil)
        (yrf nil)
        (so-symbol nil)
        (ex1 nil)
        (ex2 nil)

        )
    (x-axis (number ?a)
        (explanation1 ?c)
        (explanation2 ?d))
    (y-axis (number ?b)
          (name ?e))
     =>

     (bind ?t (remove-nil ?c ?e ?d "" "" "" ""))
       (if (numberp ?supernumber)
          then
     (assert (translation
               (translation ?t)
                    (number ?number)))
     else
          (assert (translation
                    (half-translation ?t)
                    (supernumber ?supernumber)))

     )
     )

(defrule translation-rule-b1

     (principles-for-translation2
               (supernumber ?supernumber)
          (number ?number)
          (first-priority-x1 nil)
          (first-priority-x2 nil)
          (first-priority-xr nil)
          (first-priority-y1 nil)
          (first-priority-y2 nil)
          (first-priority-yr nil)
          (x1 ?a)
          (x2 ?b)
          (x3 nil)
          (xr1  √)
          (xr2 nil)
          (xrf nil)
          (y1 ?d)
          (y2 nil)
          (y3 nil)
          (yr1 nil)
          (yr2 nil)
          (yrf nil)
          (so-symbol nil)
          (ex1 nil)
          (ex2 nil)

          )
     (x-axis (number ?a)
          (explanation1 ?e)
        )
     (x-axis (number ?b)
          (explanation1 ?f)
          (explanation2 ?g))

     (y-axis (number ?d)
          (name ?h))
     (symbol-and-meaning (symbol  √)
          (meaning1 ?i))
     =>
     (bind ?t (remove-nil ?e ?i ?f ?h "" "" ""))
```

```
        (if (numberp ?supernumber)
        then
    (assert (translation
            (translation ?t)
                (number ?number)))
    else
        (assert (translation
                (half-translation ?t)
                (supernumber ?supernumber)))
))

(defrule translation-rule-b2
    (principles-for-translation2
             (supernumber ?supernumber)
        (number ?number)
        (first-priority-x1 nil)
        (first-priority-x2 nil)
        (first-priority-xr nil)
        (first-priority-y1 nil)
        (first-priority-y2 nil)
        (first-priority-yr nil)
        (x1 ?a)
        (x2 ?b)
        (x3 nil)
        (xr1 >)
        (xr2 nil)
        (xrf nil)
        (y1 ?d)
        (y2 nil)
        (y3 nil)
        (yr1 nil)
        (yr2 nil)
        (yrf nil)
        (so-symbol nil)
        (ex1 nil)
        (ex2 nil)


        )
    (x-axis (number ?a)
        (explanation1 ?e)
        (explanation2 ?f)
       )
    (x-axis (number ?b)
        (explanation5 ?g)
       )

    (y-axis (number ?d)
        (name ?h))
    (symbol-and-meaning (symbol >)
        (meaning3 ?i))
    =>
    (bind ?t (remove-nil ?e ?h ?f ?g "" "" ""))
          (if (numberp ?supernumber)
        then
    (assert (translation
            (translation ?t)
                (number ?number)))
```

```
     else
          (assert (translation
                         (half-translation ?t)
                         (supernumber ?supernumber)))
))

(defrule translation-rule-c
     (principles-for-translation2
                (supernumber ?supernumber)
          (number ?number)
          (first-priority-x1 nil)
          (first-priority-x2 nil)
          (first-priority-xr nil)
          (first-priority-y1 nil)
          (first-priority-y2 nil)
          (first-priority-yr nil)
          (x1 ?a)
          (x2 nil)
          (x3 nil)
          (xr1 nil)
          (xr2 nil)
          (xrf nil)
          (y1 ?b)
          (y2 ?c)
          (y3 nil)
          (yr1 ?d)
          (yr2 nil)
          (yrf nil)
          (so-symbol nil)
          (ex1 nil)
          (ex2 nil)


          )
     (x-axis (number ?a)
          (explanation1 ?e)
          (explanation2 ?f)
        )
     (y-axis (number ?b)
          (name ?g)
        )

     (y-axis (number ?c)
          (name ?h)
          (adjname ?i))
     (symbol-and-meaning (symbol ?d)
          (meaning1 ?j))
     =>
     ;;(bind ?t1 (str-cat ?e " " ?i   " " ?g   " "   ?f " or " ?e " " ?g " " ?j " " ?h " " ?f))
    (bind ?t1 (remove-nil ?e ?i ?g ?f "" "" ""))
    (bind ?t2 (remove-nil ?e ?g ?j ?h ?f "" ""))
     (bind ?t (str-cat ?t1 " or " ?t2))
               (if (numberp ?supernumber)
          then
     (assert (translation
               (translation ?t)
                    (number ?number)))
     else
```

```
            (assert (translation
                        (half-translation ?t)
                        (supernumber ?supernumber)))
))

(defrule translation-rule-d

      (principles-for-translation2
                  (supernumber ?supernumber)
            (number ?number)
            (first-priority-x1 nil)
            (first-priority-x2 nil)
            (first-priority-xr nil)
            (first-priority-y1 nil)
            (first-priority-y2 nil)
            (first-priority-yr nil)
            (x1 ?a)
            (x2 ?b)
            (x3 nil)
            (xr1 ?c)
            (xr2 nil)
            (xrf nil)
            (y1 ?d)
            (y2 ?e)
            (y3 nil)
            (yr1 ?f)
            (yr2 nil)
            (yrf nil)
            (so-symbol nil)
            (ex1 nil)
            (ex2 nil)


            )
      (x-axis (number ?a)
            (explanation1 ?g)
         )
      (x-axis (number ?b)
            (explanation1 ?h))
      (y-axis (number ?d)
            (name ?i)
         )

      (y-axis (number ?e)
            (name ?j)
            (adjname ?k))
      (symbol-and-meaning (symbol ?c)
            (meaning1 ?l))
      (symbol-and-meaning (symbol ?f)
            (meaning1 ?m))
      =>
  ;;  (printout t "test" crlf)
    (bind ?t1 (remove-nil ?g ?l ?h ?i ?m ?j ""))
    (bind ?t2 (remove-nil ?g ?l ?h ?k ?i "" ""))
      (bind ?t (str-cat ?t1 " or " ?t2))
              (if (numberp ?supernumber)
            then
      (assert (translation
                   (translation ?t)
                         (number ?number)))
      else
            (assert (translation
                         (half-translation ?t)
                         (supernumber ?supernumber)))
))

(defrule translation-rule-e

      (principles-for-translation2
                    (supernumber ?supernumber)
            (number ?number)
            (first-priority-x1 ?a)
            (first-priority-x2 nil)
            (first-priority-xr nil)
            (first-priority-y1 ?b)
            (first-priority-y2 nil)
            (first-priority-yr nil)
            (x1 ?c)
            (x2 nil)
            (x3 nil)
            (xr1 nil)
            (xr2 nil)
            (xrf nil)
            (y1 nil)
            (y2 nil)
            (y3 nil)
            (yr1 nil)
            (yr2 nil)
            (yrf nil)
            (so-symbol nil)
            (ex1 nil)
            (ex2 nil)


              )
      (x-axis (number ?a)
            (explanation3 ?d)
            (explanation4 ?e)
          )
      (x-axis (number ?c)
            (explanation1 ?f)
            (explanation2 ?g))
      (y-axis (number ?b)
            (name ?h)
          )

      =>
   ;;  (printout t "test" crlf)
    (bind ?t1 (remove-nil ?f ?d ?h ?g "" "" ""))
    (bind ?t2 (remove-nil ?f ?e ?h ?g "" "" ""))
      (bind ?t (str-cat ?t1 " or " ?t2))
            (if (numberp ?supernumber)
            then
      (assert (translation
                   (translation ?t)
                         (number ?number)))
    else
        (assert (translation
                (half-translation ?t)
                (supernumber ?supernumber)))
))

(defrule translation-rule-f
    (principles-for-translation2
            (supernumber ?supernumber)
         (number ?number)
        (first-priority-x1 ?a)
        (first-priority-x2 ?b)
        (first-priority-xr ?c)
        (first-priority-y1 ?d)
        (first-priority-y2 nil)
        (first-priority-yr nil)
        (x1 ?e)
        (x2 nil)
        (x3 nil)
        (xr1 nil)
        (xr2 nil)
        (xrf nil)
        (y1 nil)
        (y2 nil)
        (y3 nil)
        (yr1 nil)
        (yr2 nil)
        (yrf nil)
        (so-symbol nil)
        (ex1 nil)
        (ex2 nil)



         )
    (x-axis (number ?a)
        (explanation5 ?f)
       )
    (x-axis (number ?b)
        (explanation5 ?g)
     )
        (x-axis (number ?e)
        (explanation1 ?h)
        (explanation2 ?i)
     )

    (y-axis (number ?d)
        (name ?j)
       )
   (symbol-and-meaning (symbol ?c)
        (meaning3 ?k))
     =>
   ;; (printout t "test" crlf)
   (bind ?t (remove-nil ?h ?j ?k ?f ?g ?i ""))
        (if (numberp ?supernumber)
        then
    (assert (translation
             (translation ?t)
                (number ?number)))
    else
        (assert (translation
                (half-translation ?t)
                (supernumber ?supernumber)))
))

(defrule translation-rule-g

    (principles-for-translation2
            (supernumber ?supernumber)
        (number ?number)
        (first-priority-x1 ?a)
        (first-priority-x2 nil)
        (first-priority-xr nil)
        (first-priority-y1 ?b)
        (first-priority-y2 ?c)
        (first-priority-yr ?d)
        (x1 ?e)
        (x2 nil)
        (x3 nil)
        (xr1 nil)
        (xr2 nil)
        (xrf nil)
        (y1 nil)
        (y2 nil)
        (y3 nil)
        (yr1 nil)
        (yr2 nil)
        (yrf nil)
        (so-symbol nil)
        (ex1 nil)
        (ex2 nil)
        )
    (x-axis (number ?a)
        (explanation1 ?f)
        (explanation4 ?g)
       )
        (x-axis (number ?e)
        (explanation1 ?h)
        (explanation2 ?i)
     )

    (y-axis (number ?b)
        (name ?j))

    (y-axis (number ?c)
        (name ?k)
       )
   (symbol-and-meaning (symbol ?d)
        (meaning1 ?l))
    =>
    ;;(printout t "test")
   (bind ?t1 (remove-nil ?h ?f ?j ?l ?k ?i ""))
   (bind ?t2 (remove-nil ?h ?g ?j ?l ?k ?i ""))
    (bind ?t (str-cat ?t1 " or " ?t2))
         (if (numberp ?supernumber)
        then
    (assert (translation
            (translation ?t)
                (number ?number)))
    else
        (assert (translation
                (half-translation ?t)
                (supernumber ?supernumber)))
))

(defrule translation-rule-i
    (principles-for-translation2
             (supernumber ?supernumber)
        (number ?number)
        (first-priority-x1 nil)
        (first-priority-x2 nil)
        (first-priority-xr nil)
        (first-priority-y1 nil)
        (first-priority-y2 nil)
        (first-priority-yr nil)
        (x1 ?a)
        (x2 nil)
        (x3 nil)
        (xr1 nil)
        (xr2 nil)
        (xrf nil)
        (y1 ?b)
        (y2 ?c)
        (y3 ?d)
        (yr1 ?e)
        (yr2 ?f)
        (yrf nil)
        (so-symbol nil)
        (ex1 nil)
        (ex2 nil)
        )
    (x-axis (number ?a)
        (explanation1 ?g)
        (explanation2 ?h)
       )


    (y-axis (number ?b)
        (name ?i))

    (y-axis (number ?c)
        (name ?j)
       )
        (y-axis (number ?d)
        (name ?k)
       )
   (symbol-and-meaning (symbol ?e)
        (meaning1 ?l))
       (symbol-and-meaning (symbol ?f)
        (meaning1 ?m))
    =>
    ;;(printout t "test")
   (bind ?t (remove-nil ?g   ?i ?l ?j ?m ?k ?h))
           (if (numberp ?supernumber)
        then
    (assert (translation
            (translation ?t)
                (number ?number)))
    else
        (assert (translation
                (half-translation ?t)
                (supernumber ?supernumber)))
))

(defrule translation-rule-j

    (principles-for-translation2
              (supernumber ?supernumber)
        (number ?number)
        (first-priority-x1 nil)
        (first-priority-x2 nil)
        (first-priority-xr nil)
        (first-priority-y1 nil)
        (first-priority-y2 nil)
        (first-priority-yr nil)
        (x1 ?a)
        (x2 nil)
        (x3 nil)
        (xr1 nil)
        (xr2 nil)
        (xrf nil)
        (y1 ?b)
        (y2 ?c)
        (y3 ?d)
        (yr1 ?e)
        (yr2 nil)
        (yrf ?f)
        (so-symbol nil)
        (ex1 nil)
        (ex2 nil)
        )
    (x-axis (number ?a)
        (explanation1 ?g)
        (explanation2 ?h)
      )


    (y-axis (number ?b)
        (name ?i))

    (y-axis (number ?c)
        (name ?j)
      )
        (y-axis (number ?d)
        (name ?k)
      )
   (symbol-and-meaning (symbol ?e)
        (meaning1 ?l))
      (symbol-and-meaning (symbol ?f)
        (meaning1 ?m))
   =>
   ;;(printout t "test")
   (bind ?t (remove-nil ?g ?i ?l ?j ?m ?k ?h))
```

```
         (if (numberp ?supernumber)
        then
    (assert (translation
            (translation ?t)
                (number ?number)))
    else
        (assert (translation
                (half-translation ?t)
                (supernumber ?supernumber)))
))


(defrule translation-rule-k
    (principles-for-translation2
             (supernumber ?supernumber)
        (number ?number)
        (first-priority-x1 nil)
        (first-priority-x2 nil)
        (first-priority-xr nil)
        (first-priority-y1 nil)
        (first-priority-y2 nil)
        (first-priority-yr nil)
        (x1 ?a)
        (x2 ?b)
        (x3 ?c)
        (xr1 ?d)
        (xr2 nil)
        (xrf ?e)
        (y1 ?f)
        (y2 nil)
        (y3 nil)
        (yr1 nil)
        (yr2 nil)
        (yrf nil)
        (so-symbol nil)
        (ex1 nil)
        (ex2 nil)
        )
    (x-axis (number ?a)
        (explanation1 ?g)
        (explanation2 ?h)
       )

        (x-axis (number ?b)
        (explanation1 ?i)

       )

            (x-axis (number ?c)
        (explanation1 ?j)

       )


    (y-axis (number ?f)
        (name ?k))
    (symbol-and-meaning (symbol ?d)
         (meaning2 ?l))
        (symbol-and-meaning (symbol ?e)
         (meaning1 ?m))
     =>
    ;;(printout t "test")
    (bind ?t (remove-nil ?g ?k ?h ?l ?i ?m ?j))
          (if (numberp ?supernumber)
         then
     (assert (translation
             (translation ?t)
                  (number ?number)))
     else
         (assert (translation
                  (half-translation ?t)
                  (supernumber ?supernumber)))
))

;;(defmodule sixth-part)

(defrule output-single-result
     (translation (translation ?t1)(number ?number))
     =>
     ;;(assert               (temp-translation               (number               ?number)
(translation1 ?t1)(translation2 ?t2)(translation3 ?t3)(translation4 ?t4)))

     (bind ?t' (str-cat "
" ?t1 ?*crlf* ))
             (?*qfield* append ?t')
       )

(defrule output-doubleiresult
     (translation (half-translation ?t1&: (neq ?t1 nil))(supernumber ?supernumber))
     (translation (half-translation ?t2&: (neq ?t1 nil))(supernumber ?supernumber))
     =>
     (bind ?t (str-cat ?t1 "in oder to" ?t2 ?*crlf*
                        ?t1 "according to" ?t2))
      (?*qfield* append ?t)
     )
```

# Appendix D. Code for conflict B resolution module

```
(defmodule conflict-detectionB0)

(defrule conflictA-existed
      (trigger-collaboration (id 3))
      =>
    (ask-user-if-continue-after-conflictA "
Recommendations about resolving ConflictA are shown above,
choose yes if you want to continue...
choose no if you want to quit....")
       ((engine) waitForActivations)
       )

(defrule conflictA-nonexisted
      (trigger-collaboration (id 2))
      (not (trigger-collaboration (id 3)))
      =>
          (ask-user-if-continue-after-conflictA "
There is no conflictA in this problem,
choose yes if you want to continue...
choose no if you want to quit....")
             ((engine) waitForActivations)
      )


(defmodule conflict-detectionB1)

(defrule conflict-B-detection-rule
      (user-input-collaboration (text yes))
    (MAIN::Goal (weight ?w1) (T-parameter ?t1)(name ?n1))
    (MAIN::Goal (weight ?w2) (T-parameter ?t2)(name ?n2))
      (MAIN::final-contradiction-constraint-with-cell
           (first-objective ?t1)
           (second-objective ?t2))


      =>
  ;;  (retract ?a)
           (bind ?w (* ?w1 ?w2))
       (if (< ?w 90)
           then
          (assert                       (conflict-information                        (catelgoue
B)(first-t ?t1)(second-t ?t2)(first-goal ?n1)(second-goal ?n2)(weight ?w)))
          (assert (conflictb-nonrepeated (constraint1 ?t1)(constraint2 ?t2)))
            (assert (trigger-collaboration (id 11)))
            (assert (trigger-collaboration (id 12)))
            (assert (conflictA-and-B (t1 ?t1) (t2 ?t2)))
          (?*qfield* setText "Conflict B is shown as follows:
")



           )
      )
```

```
(defmodule conflict-detectionB1-2)

(defrule show-conflictb
      (trigger-collaboration (id 12))
  (conflictb-nonrepeated (constraint1 ?t1)(constraint2 ?t2))
      (MAIN::objectives (number ?t1) (name ?c))
      (MAIN::objectives (number ?t2) (name ?d))
     ;;(MAIN::Goal (T-parameter ?t1)(name ?n1))
     ;;(MAIN::Goal   (T-parameter ?t2)(name ?n2))
          =>
           (bind ?s "")
      (bind ?s (str-cat   ?c " (" ?t1 ") "     "Vs " ?d " ("   ?t2 ") "   ?*crlf* ))
      (?*qfield* append ?s)

        ;;(facts)
     )

(defmodule conflict-detectionB2)

(defrule ask-if-show-principles
         (trigger-collaboration (id 11))
      =>
          (ask-user-if-show-principles)
          ((engine) waitForActivations)
   ;;   (assert (trigger-collaboration (id 7)))
      )

(defrule show-princiles
     ;; (trigger-collaboration (id 7))
      (MAIN::another-yes-or-no (text yes))
      (conflict-information                                          (catelgoue
B)(first-t ?t1)(second-t ?t2)(first-goal ?n1)(second-goal ?n2)(weight ?w))
     (MAIN::final-contradiction-constraint-with-cell (first-objective ?t1) (second-objective ?t2)
(cell $?cell))
      =>
     ;; (printout t "Test")
     (bind ?*celllist* (create$ ?*celllist* ?cell))
   ;;(assert (templist (number ?cell) (mark ?w)))
       (foreach ?a ?cell
          (assert (weight-for-each-principle (principle ?a) (weight ?w)))
      )
;;   (bind ?*mark* (+ ?*mark* 1))
 ;; (bind ?*mark1* (- ?*mark* 1))
   (assert (trigger-collaboration (id 6)))
   ;;(printout t ?*celllist*)
  ;;    (facts)
      )

(defmodule conflict-detectionB3)

 (defrule calculate-weight-and-frequency
      (weight-for-each-principle(principle ?p) (weight ?w))
      (frequency-of-principles (frequency ?n)(principles ?p))
 ;;                  (MAIN::principles          (number          ?p)          (text          ?text1)
(symbol-expression1 ?e1)(symbol-expression2 ?e2)(symbol-expression3 ?e3)(symbol-expressi
on4 ?e4) (symbol-expression5 ?e5))
```

```
        =>
        ;;(printout t "test")
             (bind ?nw' (* ?w ?n))
       (bind ?nw (integer ?nw'))

             (assert (frequency-of-principles (weight ?nw)(principles ?p) (frequency ?n)))
           ;; (facts)
                        (bind ?s "Principles are shown by the importance from large to small")
        (?*qfield* setText ?s)
     ;; (facts)
      )

(defmodule conflict-detectionB4-1)
(defrule print-frequency-and-principles-1000
       (frequency-of-principles (weight ?nw)(principles ?p) (frequency ?n))
        (MAIN::principles            (number            ?p)            (text            ?text1)
(symbol-expression1 ?e1)(symbol-expression2 ?e2)(symbol-expression3 ?e3)(symbol-expressi
on4 ?e4) (symbol-expression5 ?e5))
       =>
       (if (and (neq ?nw nil)(> ?nw 5000))
            then
   ;; (bind ?m (+ ?n 1))
       (bind ?x "
The principle No. ")
       (bind ?x (str-cat ?x ?p " appears " ?n " times. " ?*crlf*))
       (bind ?x (str-cat ?x "The possiblity of solving problems by using this principle is
" ?nw ?*crlf*))
       (bind ?x (str-cat ?x "Meaning: " ?text1 ?*crlf*))
       (bind ?x (str-cat ?x "Expressions: " ?*crlf*))
       (bind ?x1 (remove-nil-crlf ?e1 ?e2 ?e3 ?e4 ?e5))
       (bind ?x (str-cat ?x ?x1 ?*crlf* "-------------------" ?*crlf*))
            ;;(bind ?s "")
       ;;(?*qfield* setText ?x)
        (?*qfield* append ?x)
             )
        ;;(printout t "The principle No. " ?p " appears " ?m " times." crlf)
)
```

# Appendix E. Code for conflict C resolution module

```
(defmodule conflict-detectionC)

(defrule conflictB-nonexisted
     (trigger-collaboration (id 2))
     (not (trigger-collaboration (id 12)))
     =>
          (ask-user-if-continue-after-conflictB "
There is no conflictB in this problem,
choose yes if you want to continue...
choose no if you want to quit.... ")
             ((engine) waitForActivations)

     )

(defmodule conflict-detectionC1)

(defrule conflictc-detection-rule0
               ;; (conflictA-and-B (t1 ?t3) (t2 ?t4))
                (user-input-collaboration10 (text yes))
                (MAIN::Goal     (T-parameter     ?t1)(name     ?n1)(weight     ?w1)(party
A)(direction ?d1))
                (MAIN::Goal     (T-parameter     ?t2)(name     ?n2)(weight     ?w2)(party
B)(direction ?d2))



          =>
  (if (eq ?t1 ?t2)
         then
         (assert                    (conflict-information                    (catelgoue
C)(direction1 ?d1)(direction2 ?d2)(first-t ?t1)(second-t ?t2)(weight1 ?w1)(weight2 ?w2)(first-
goal ?n1)(second-goal ?n2)))
     (assert (trigger-collaboration (id 15)))
     (?*qfield* setText "")
          )
          ;;  (printout t "ceshi1------")
  ;;  (facts)
    )

(defmodule conflict-detectionC1-2)

(defrule ask-user-goal-extra
     (conflict-information                                        (catelgoue
C)(direction1 ?d1)(direction2 ?d2)(first-t ?t1)(second-t ?t2)(weight1 ?w1)(weight2 ?w2)(first-
goal ?n1)(second-goal ?n2))
     (trigger-collaboration (id 15))
     =>
     (ask-user-extra-goal-info ?n1 ?n2 ?d1 ?d2)
          ((engine) waitForActivations)
    (assert (trigger-collaboration (id 16)))
     )

(defrule assert-integrated-info
   ;; (trigger-collaboration (id 16))
     (conflict-information (catelgoue
C)(direction1 ?d1)(direction2 ?d2)(first-t ?t1)(second-t ?t2)(weight1 ?w1)(weight2 ?w2)(first-
goal ?n1)(second-goal ?n2))
     (temp-goal-extra
(compromisename ?comname)(alternatename ?altname)(compromise ?com)(alternate ?alt)(L
ocal/global1 ?lg1) (Local/global2 ?lg2)(authorityweight1 ?aw1)
(authorityweight2 ?aw2)(catelogue1 ?c1) (catelogue2 ?c2)
(clientweight1 ?cw1)(clientweight2 ?cw2) (goal1 ?n1) (goal2 ?n2) (subgoal1 ?sg1)
(subgoal2 ?sg2) (voteweight1 ?vw1)(voteweight2 ?vw2))
     =>
     (assert (complete-conflict-information (catelgoue
C)(compromisename ?comname)(alternatename ?altname)(compromise ?com)(alternate ?alt)
(direction1 ?d1)(direction2 ?d2)(first-t ?t1)(second-t ?t2)(weight1 ?w1)(weight2 ?w2)(first-go
al ?n1)(second-goal ?n2)(Local/global1 ?lg1) (Local/global2 ?lg2)(authorityweight1 ?aw1)
(authorityweight2 ?aw2) (catelogue1 ?c1) (catelogue2 ?c2)
(clientweight1 ?cw1)(clientweight2 ?cw2) (subgoal1 ?sg1) (subgoal2 ?sg2)
(voteweight1 ?vw1)(voteweight2 ?vw2)))
;;  (facts)
     )

(defmodule conflict-dectionC1-2-1)

(defrule ask-user-if-contiue-after-conflictc-input
     (trigger-collaboration (id 16))
     =>
     (ask-user-if-continue-after-conflictc-input)
     ((engine) waitForActivations)    )

(defmodule conflict-detectionC2)

(defrule conflictc-detection-rule1
     (yes-for-acombo11 (text yes))

 (complete-conflict-information
(catelgoue C)
(compromisename ?comname)
(alternatename ?altname)
(compromise ?com)
(alternate ?alt)
(direction1 ?d1)
(direction2 ?d2)
(first-t ?t1)
(second-t ?t2)
(weight1 ?w1)
(weight2 ?w2)
(first-goal ?n1)
(second-goal ?n2)
(Local/global1 ?lg1)
(Local/global2 ?lg2)
(authorityweight1 ?aw1)
(authorityweight2 ?aw2)
(catelogue1 ?c1)
(catelogue2 ?c2)
```

```
(clientweight1 ?cw1)
(clientweight2 ?cw2)
(subgoal1 ?sg1)
(subgoal2 ?sg2)
(voteweight1 ?vw1)
(voteweight2 ?vw2))
        =>
(bind ?abandong1 (str-cat "Abandon Goal1: " ?n1))
(bind ?abandong2 (str-cat "Abandon Goal1: " ?n2))
(bind ?alt' (str-cat "Try alternate plan: " ?altname))

(bind ?sg1' (str-cat "Try alternate subgoal " ?sg1))
(bind ?sg2' (str-cat "Try alternate subgoal " ?sg2))

;;subgoal
(if (neq ?sg1 nil)
        then
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?sg1')(value
80)(party "Design parties"))))
(if (neq ?sg2 nil)
        then
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?sg2')(value
80)(party "Design parties"))))

;;alternate
(if (eq ?alt "Design parties")
         then
         (assert (recommendation (first-goal ?n1)(second-goal ?n2) (name ?alt') (value
100)(party "Design parties")))
        )
(if (eq ?alt "Client")
        then
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?alt') (value 80)
(party "Client")))
        )
(if (eq ?alt "Other related parties")
        then
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?alt') (value
60)(party "Other related parties")))
        ;; (printout t "test")
        )

(if (eq ?alt "Authority")
        then
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?alt') (value
40)(party "Authority")))
        )

;;first condition
(if (or
        (and  (eq ?n1 ?n2) (neq ?d1 ?d2))
        (and (neq ?n1 ?n2) (eq ?d1 ?d2))
            )
        then


        ;;COMPROMISE
        (bind ?com' (str-cat "Do two-side compromise at point: " ?comname))
        (if (eq ?com "Design parties")
         then
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?com') (value
60)(party "Design parties")))
        )
        (if (eq ?com "Client")
        then
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?com') (value 48)
(party "Client")))
        )
        (if (eq ?com "Other related parties")
        then
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?com') (value
36)(party "Other related parties")))

       )

        (if (eq ?com "Authority")
        then
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?com') (value
24)(party "Authority")))
        )

         ;;abandon
        (if (> ?w1 ?w2)
        then
            (bind ?vdw (integer (* (/ (abs (- ?w1 ?w2)) ?w1) 100)))
       (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong1)
(value ?vdw)(party "Design parties")))
        )
        (if (< ?w1 ?w2)
        then
            (bind ?vdw (integer(* (/ (abs (- ?w1 ?w2)) ?w1) 100)))
       (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong2)
(value ?vdw)(party "Design parties")))
        )
        (if (> ?cw1 ?cw2)
        then
            (bind ?vcw (integer(* (/ (abs (- ?cw1 ?cw2)) ?cw1) 80)))
       (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong1)
(value ?vcw)(party "Client")))
        )
        (if (< ?cw1 ?cw2)
        then
            (bind ?vcw (integer(* (/ (abs (- ?cw1 ?cw2)) ?cw1) 80)))
       (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong2)
(value ?vcw)(party "Client")))
        )
       (if (> ?vw1 ?vw2)
        then
       (bind ?vvw (integer(* (/ (abs (- ?vw1 ?vw2)) ?vw1) 60)))
       (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong1)
(value ?vvw)(party "other related parties")))
        )
       (if (< ?vw1 ?vw2)
        then
       (bind ?vvw (integer(* (/ (abs (- ?vw1 ?vw2)) ?vw1) 60)))
       (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong2)
(value ?vvw)(party "Other related parties")))
            )
        (if (> ?aw1 ?aw2)
          then
        (bind ?vaw (integer(* (/ (abs (- ?aw1 ?aw2)) ?aw1) 40)))
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong1)
(value ?vaw)(party "Authority")))
            )
        (if (< ?aw1 ?aw2)
          then
        (bind ?vaw (integer(* (/ (abs (- ?aw1 ?aw2)) ?aw1) 40)))
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong2)
(value ?vaw)(party "Authority")))
            )
          )
(if (and (eq ?c1 ?c2) (neq ?n1 ?n2) (neq ?d1 ?d2) (eq ?lg1 Global) (eq ?lg2 Local))
          then

          ;;abandon
          (bind ?gw1 (* ?w1 80%))
          (bind ?gw2 (* ?w2 20%))
          (bind ?gcw1 (* ?cw1 80%))
          (bind ?gcw2 (* ?cw2 20%))
          (bind ?gvw1 (* ?vw1 80%))
          (bind ?gvw2 (* ?vw2 20%))
          (bind ?gaw1 (* ?aw1 80%))
          (bind ?gaw2 (* ?aw2 20%))
          (if (> ?gw1 ?gw2)
          then
          (bind ?gvdw (integer (* (/ (abs (- ?gw1 ?gw2)) ?gw1) 100)))
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong1)
(value ?gvdw)(party "Design parties")))
            )
          (if (< ?gw1 ?gw2)
          then
        (bind ?gvdw (integer(* (/ (abs (- ?gw1 ?gw2)) ?gw1) 100)))
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong2)
(value ?vdw)(party "Design parties")))
            )
        (if (> ?gcw1 ?gcw2)
          then
          (bind ?gvcw (integer(* (/ (abs (- ?gcw1 ?gcw2)) ?gcw1) 80)))
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong1)
(value ?gvcw)(party "Client")))
            )
        (if (< ?gcw1 ?gcw2)
          then
        (bind ?gvcw (integer(* (/ (abs (- ?gcw1 ?gcw2)) ?gcw1) 80)))
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong2)
(value ?gvcw)(party "Client")))
            )
        (if (> ?gvw1 ?gvw2)
          then
        (bind ?gvvw (integer(* (/ (abs (- ?gvw1 ?gvw2)) ?gvw1) 60)))
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong1)
(value ?gvvw)(party "other related parties")))
            )
        (if (< ?gvw1 ?gvw2)
        then
       (bind ?gvvw (integer(* (/ (abs (- ?gvw1 ?gvw2)) ?gvw1) 60)))
       (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong2)
(value ?gvvw)(party "Other related parties")))
         )
       (if (> ?gaw1 ?gaw2)
        then
       (bind ?gvaw (integer(* (/ (abs (- ?gaw1 ?gaw2)) ?gaw1) 40)))
       (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong1)
(value ?gvaw)(party "Authority")))
         )
       (if (< ?gaw1 ?gaw2)
        then
       (bind ?gvaw (integer(* (/ (abs (- ?gaw1 ?gaw2)) ?gaw1) 40)))
       (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?abandong2)
(value ?gvaw)(party "Authority")))
         )

         ;;COMPROMISE
       (bind ?onesidecom' (str-cat "Do compromise on Goal 2 " ?n2  " at point:
" ?comname))
       (if (eq ?com "Design parties")
        then
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?onesidecom')
(value 60)(party "Design parties")))
         )
        (if (eq ?com "Client")
        then
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?onesidecom')
(value 48) (party "Client")))
         )
        (if (eq ?com "Other related parties")
        then
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?onesidecom')
(value 36)(party "Other related parties")))
        )

        (if (eq ?com "Authority")
        then
        (assert (recommendation (first-goal ?n1)(second-goal ?n2)(name ?onesidecom')
(value 24)(party "Authority")))
         )
)
     (assert (trigger-collaboration (id 17)))
     (?*qfield* setText "
*****************************************************
Following are recommendations for resolving conflictC
*****************************************************
"))

(defmodule conflict-detectionC3-80)

(defrule conflictc-resolution-display
     (recommendation (name ?n) (value ?v&: (> ?v 80))(party ?p)(first-goal ?n1)
(second-goal ?n2))
     (trigger-collaboration (id 17))
     =>
     (bind ?s (str-cat "
The resolution of conflict C " ?n1 " and " ?n2 " is:
" ?n "
Weight of this resolution is :" ?v "
Suggested by: " ?p "
-------------------------
"))
    (?*qfield* append ?s)
    )

(defmodule conflict-detectionC3-60)

(defrule conflictc-resolution-display
    (recommendation (name ?n)(value ?v&: (and (> ?v 60) (<= ?v
80)))(party ?p)(first-goal ?n1)
(second-goal ?n2))
    (trigger-collaboration (id 17))
    =>
    (bind ?s (str-cat "

The resolution of conflict C " ?n1 " and " ?n2 " is:
" ?n "
Weight of this resolution is :" ?v "
Suggested by: " ?p "
-------------------------
"))
    (?*qfield* append ?s)
    )

(defmodule conflict-detectionC3-40)

(defrule conflictc-resolution-display
    (recommendation (name ?n) (value ?v&: (and (> ?v 40) (<= ?v
60)))(party ?p)(first-goal ?n1)
(second-goal ?n2))
    (trigger-collaboration (id 17))
    =>
    (bind ?s (str-cat "

The resolution of conflict C " ?n1 " and " ?n2 " is:
" ?n "
Weight of this resolution is :" ?v "
Suggested by: " ?p "
-------------------------
"))
    (?*qfield* append ?s)
    )

(defmodule conflict-detectionC3-20)

(defrule conflictc-resolution-display
    (recommendation (name ?n) (value ?v&: (and (> ?v 20) (<= ?v
40)))(party ?p)(first-goal ?n1)
(second-goal ?n2))
    (trigger-collaboration (id 17))
    =>
    (bind ?s (str-cat "
The resolution of conflict C " ?n1 " and " ?n2 " is:
" ?n "
Weight of this resolution is :" ?v "
Suggested by: " ?p "
--------------------------
"))
      (?*qfield* append ?s)
      )

(defmodule conflict-detectionC3-0)

(defrule conflictc-resolution-display
      (recommendation   (name   ?n)   (value   ?v&:   (and   (>   ?v   0)   (<=   ?v
20)))(party ?p)(first-goal ?n1)
(second-goal ?n2))
      (trigger-collaboration (id 17))
      =>
      (bind ?s (str-cat "

The resolution of conflict C " ?n1 " and " ?n2 " is:
" ?n "
Weight of this resolution is :" ?v "
Suggested by: " ?p "
--------------------------
"))
      (?*qfield* append ?s)
      )

(defmodule conflict-detectionC4)

(deffunction ask-user-if-continue-after-conflictC(?s)
     (?*apanel* removeAll)
        (?*apanel* add ?*scroll*)
     (?*qfield* append ?s)
     ;; (?*apanel* add ?*alabel9*)
      (?*apanel* add ?*acombo200*)
      (?*apanel* add ?*acombo-ok200*)
     ;; ((?*frame-c* getContentPane) add ?*apanel* (get-member BorderLayout SOUTH))
      ((?*frame-c* getContentPane) add ?*apanel* )
      (?*frame-c* validate)
       (?*frame-c* repaint)
          )

(defrule conflictC-existed
      (trigger-collaboration (id 15))
      =>
    (ask-user-if-continue-after-conflictC "
Recommendations about resolving ConflictC are shown above,
choose yes if you want to continue...
")
       ((engine) waitForActivations)
       )

(defrule conflictC-nonexisted
      (trigger-collaboration (id 2))
      (not (trigger-collaboration (id 15)))
      =>
         (ask-user-if-continue-after-conflictC "
There is no conflictC in this problem,
choose yes to quit...
")
            ((engine) waitForActivations))
```

# Reference

Adelson B., 1999, “Developing strategic alliances: A framework for collaborative negotiation in design”, Research in Engineering Design (11): pp. 133-44.

Alberts, L. K., 1994, “YMIR: A sharable ontology for the formal representation of engineering design knowledge”, IFIP Transactions B: Computer Applications in Technology B (18): pp. 3-32.

Altshuller, G., 1988, Translated by A. Williams. “Creativity as an Exact Science”, Gordon and Breach, New York.

Altshuller, G., 1996, “And suddenly the inventor appeared : TRIZ, the theory of inventive problem solving”, Worcester, MA : Technical Innovation Center, Inc.

Al-Hakim L, K. A., Mathew J., 2000, “A graph-theoretic approach to conceptual design with functional perspectives”, Computer aided design 32: pp. 867-75.

Al-Salka MA, C. M., Hardy SJ, 1998, “A framework for a generalised computer-based support environment for conceptual engineering design”, Journal of Engineering Design 9(1): pp. 57-88.

Andersson, K., Makkonen, P., Persson, J. G., 1995, “A Proposal to a Product Modeling Language to support Conceptual Design”, Annals of CIRP, Vol. 44, pp. 129-132.

Barry O., James B., 1998, “A constraint-based approach to supporting conceptual design”, In John Gero and Fay Sudweeks, editors, Artificial Intelligence in Design '98, pp. 291-308, The Netherlands, July 1998. Kluwer Academic Publishers.

Bentley, P.J., Wakefield., J.P., 1997, "Conceptual evolutionary design by genetic algorithms", Engineering Design and Automation, 3(2): pp. 119-131.

Bentley, P., 1996, "Generic Evolutionary Design of Solid Objects using a Genetic Algorithm", PhD thesis, University of Huddersfield.

BOS, A.H.W., 1998, "Aircraft conceptual design by genetic/gradient-guided optimisation", Engineering Applications of Artificial Intelligence, 11: pp. 377-382.

Bracewell, R., 2002, "Synthesis based on function - means trees: Schemebuilder", In Engineering Design Synthesis (Chakrabarti, A., Ed.) pp. 199 - 212.

Brunetti G, G. B., 2000, "A feature-based approach towards an integrated product model including conceptual design information", Computer aided design 32: pp. 877-87.

Buzan, A., 1993, "The Mind Map Book", London, BBC Books.

Castelfranchi, C., 2000, "Conflict ontology", In Computational Conflicts, Conflict Modelling for Distributed Intelligent Systems, edited by H.J. Mu¨ ller and R. Dieng, pp. 21–40, (Springer).

Charles J. Petrie, Teresa A. Webster, and Mark R. Cutkosky, 1995, "Using Pareto optimality to coordinate distributed agents", AIEDAM, 9: pp. 269-281.

Campbell MI, C. J., Kotovsky K., 1999, "A-Design: an agent-based approach to conceptual design in a dynamic environment", Research in Engineering Design 11: pp. 172-192.

Cedrone, M. J., 2004, "Using a Negotiations Lens to Examine the American Catholic Church's Response to the Clergy Sex-Abuse Scandal." Negotiation Journal on the process of dispute settlement. 20(1), pp. 65-77.

Cooper S, Taleb-Bendiab A., 1998, "CONCENSUS:multi-party negotiation support for conflict resolution in concurrent engineering design", Journal of Intelligent Manufacturing (9): pp. 155-9.

David, E.G, 1989, "Genetic Algorithms in Search, Optimization and Machine Learning", Addison-Wesley, Reading, MA.

De Bono, E., 1970, "Lateral Thinking: Creativity Step by Step", Harper and Row Publishers, New York.

Deng Y.M., 2000, "Constraint-based functional design verification for conceptual design", Computer aided design 32: pp. 889-99.

Dyer, M.G., Flowers, M., Hodges, J., 1986, "EDISON: An engineering design invention system operating naively". INT. J. ARTIFICIAL INTELLIG. Vol. 1, no. 1, pp. 36-44.

Engels, F., 1968, "Dialectics of Nature", New York, Intl Pub.

Eschenaure, H., Koski, J., Osycka, A., 1990, "Multi-criteria Design Optimisation", New York Springer-Verlag.

Fey, V.R., Rivin, E.I., 2005, "Innovation on Demand: New Product Development Using TRIZ", Cambridge, Cambridge University Press.

Finger S, Dixon J R., 1989, "A review of research in mechanical engineering design. Part 1:descriptive, prescriptive, and computer-based model, model of design process [J]", Res in Eng Design, 1: pp. 51-67.

Forbus K.D., 1988, In: Shrobe H, editor. "Qualitative physics: past, present, and future, exploring artificial intelligence", Morgan Kaufmann, San Francisco, pp. 239-96

French, M., J, 1985, "Conceptual Design for engineers", London, Design Council.

Fruchter, R., Law, K.H., Qstruc, Y.I., 1991, "An approach for qualitative structural analysis", In the Second International Conference on the Application of Artificial Intelligence to Civil and Structural Engineering. Civil-Comp Press.

Gero, J.S., Louis, S., 1995, "Improving pareto optimal designs using genetic algorithms", Microcomputers in Civil Engineering, 10(4): pp. 241-249.

Hegel, G.W.F., 1896, Lectures on the History of Philosophy. London. Kegan Paul.

Huang, G..Q., Brandon, J.A., 1993, "Agents for cooperating expert systems in concurrent engineering design", Artificial Intelligence for Engineering Design, Analysis, and Manufacturing, 7, pp. 145–158.

Hsu, W.L.B., 2000, "Conceptual design: issues and challenges", Computer Aided Design 32: pp. 849-850.

Hsu, C.C., Ho, C.S., 2004, "A New Hybrid Case-Based Architecture for Medical Diagnosis," Information Sciences - An International Journal, Vol. 166, No. 1-4, pp. 231-247.

Hyman, B. 1998, “Fundamentals of engineering design,” New Jersey, Prentice-Hall, Inc.

Hyman, B., 1998, “Fundamentals of Engineering Design”, Prentice-Hall: Englewood Cliffs, New Jersey.

Huang G.Q., 1999, “Web-based morphological charts for concept design in collaborative product development”, Journal of Intelligent Manufacturing 10: pp. 267-78.

Kawakami, H., Katai, O., Sawaragi, T., Iwai, S., 1996, “Knowledge acquisition method for conceptual design based on value engineering and axiomatic design theory”, Artifical Intelligence in Engineering, 1: pp. 187-202.

Klein, M, 1991, “Supporting conflict resolution in cooperative design systems”, IEEE Trans. Syst., Man. Cybernetics, 21(6), pp. 1379–1390.

Klein, M., 2000, “Towards a systematic repository of knowledge about managing collaborative design conflicts”, In Artificial Intelligence in Design’00, edited by J. Gero, pp. 129–146 (Kluwer Academic Publishers: Boston, MA).

Khajehpour, S. & Grierson, D.E., 1999, “Filtering of Pareto-Optimal trade-off surfaces for building conceptual design”, in Topping B.H.V. & Kumar, B. (eds) Optimzation & control in Civil & Structural Engineering, Civil-Comp Press, Edinburgh UK, pp. 63-70.

Labovitz, G. H., 1980, Managing conflict. Business Horizons, June, pp. 30–37.

Lander, S.E., 1997, “Issues in multi-agent design system”, IEEE Expert, 12(March/April), pp. 18–26.

Lara, M.A., 1999, Conflict resolution in collaborative facility design, PhD dissertation, Purdue University, West Lafayette.

Miles, L.D., 1965, "Techniques of value analysis", New York, McGraw Hill.

Levary, R.R., 1988, "Engineering design: better results through operations research methods", North-holland publishing Co.

Lidsky, D.B., 1998, "The conceptual-level Design Approach to Complex systems", Engineering-Electrical Engineering and Computer Sciences. Berkeley, University of California, Berkeley.

Li, C.L., Tan S.T., Chan, K.W., 1996, "A qualitative and heuristic approach to the conceptual design of mechanisms". Engineering application of Artificial Intelligence. 9(1), pp. 17–31.

Mann, D. L., 2002, "Hands-on Systematic Innovation", Belgium, CREAX.

Mann, D.L., Dewulf, S., Zlotin, B., Zusman, A., 2003, "Matrix 2003: Updating the TRIZ Contradiction Matrix", CREAX Press, Belgium.

Marefat, M, Malhotra S, Kashyap R.L., 1993, "Object-oriented intelligent computer-integrated design, process planning, and Inspection[J]", COMPUTER, pp. 17-20.

Matthew I., Campbell, J.C, and Kenneth K., 1998, "A-design: Theory and implementation of an adaptive agent-based method of conceptual design", In John Gero and Fay Sudweeks, editors, Artificial Intelligence in Design '98, The Netherlands, July 1998. Kluwer Academic Publishers, pp. 579-598, McKim, Robert H., 1980, "Experiences in Visual Thinking by Robert H", 2nd edition, Brooks/Cole Pub Co.

McNeil T, G. J., Warren J., 1998, "Understanding conceptual electronic design using protocol analysis", Research in Engineering Design 10: pp. 129-40.

Michael G.H., Parmee, Ian C., 1995, "The application of genetic algorithms to conceptual design", In John Sharpe, editor, AI Systems Support for Conceptual design -Proceedings of the Lancaster Internation Workshop on Engineering Design 1995, Berlin, March 1995. Springer-Verlag. 3-540-76000-8, pp. 17-36,

Navichandra, D., 1991, "Exploration and innovation in design: towards a computational model," Springer-Verlag, New York. NY, US.

Nakagawa, T., 2001, "Learning and Applying the Essence of TRIZ with Easier USIT Procedure", ETRIA World Conference: TRIZ Future 2001, Nov. 7-9, 2001, Bath, UK, pp. 151-164 .

Nakagawa, T., 2002, "Reorganizing TRIZ Solution Generation Methods into Simple Five in USIT", ETRIA World Conference: TRIZ Future 2002, Strasbourg, France, Nov. 6-8, 2002, pp. 333-345

Nakagawa, T., 2001, "Essence of TRIZ in 50 words", The TRIZ Journal, http://www.triz-journal.com/archives/2001/06/d/index.htm, last access date: 14, Jul 07.

Osborn, A.F., 1993, "Applied Imagination: Principles and Procedures of Creative Problem Solving", Hadley, MA, CEF Press.

O'Sullivan, B., 1998, "Conflict management and negotiation for Concurrent

Engineering using Pareto optimality", In R. Mackay N. Martensson and S. Bjorgvinsson, editors, Changing the ways we work-Shaping the ICT-solutions for the next century, Advances in Design and Manufacturing, pages 359-368, Amsterdam, IOS Press. Proceedings of the Conference on Integration in Manufacturing, Goteborg, Sweden.

O'Sullivan, B., 2002, "Constraint-Aided Conceptual Design", London, Professional Engineering Publishing Limited.

Owen, W.F., 1985, "Metaphor analysis of cohesiveness in small discussion groups. Small Group Behaviour", 16(3), pp. 415–424.

Pahl, G., W. b., 1995, "Engineering Design: A Systematic Approach", London, Springer-Verlag.

Parmee, I.C., 1994, "Adaptive search techniques for decision support during preliminary engineering design", In Proceedings of Informing Technologies to Support Engineering Decision Making, EPSRC/DRAL Seminar, Institution of Civil Engineers, London.

Paynter, H.M., 1961, "Analysis and design of engineering systems. Cambridge", MA:MIT Press.

Pham, D.T., Liu, H., 2006, "I-Ching-TRIZ inspired tool for retrieving conceptual design solutions", Proc. 2nd Virtual International Conference on Intelligent Production Machines and Systems, http://conference.iproms.org (July 2006)

Proctor T, 1997, "New development in computer assisted creative problem solving, Creativity and Innovation Management", Vol. 6, No. 2, pp. 94-98(5).

Quinsan C., 2002, "Generative Design: Rule-Based Reasoning in Design Process", International Conference on Generative Art, 2002, Milan. (Available at http://www.generativeart.com/papersGA2002/41.htm, last accessed: 03 April, 2006)

Rafiq, M.Y., Bugmann, G. & Easterbrook, D.J., 1999, "Building concept generation using genetic algorithms integrated with neural networks", In Borkowski, A. (Ed.) AI in Structural Eng.: IT for Design, Manufacturing, Maintenance and Monitoring. Proceedings of the 6th EG-SEA-AI Workshop, Wierzba, Poland, pp. 165-173.

Raymond V.O., 1971, "I ching", based on the translation by James Legge New York : New American Library.

Raven, A.D., 1971, "Profit improvement by value analysis, value engineering and purchase price analysis", London, Cassell.

Sam, C., 2001, "Case study: The application of TRIZ to economy class aircraft cabin design", The TRIZ Journal.http://www.triz-journal.com/archives/2001/12/f/, last access date: 14 Jul. 07.

Sabouni, A.R., Al-Mourad, O.M., 1997, "Quantitative knowledge based approach for preliminary design of tall buildings", Artificial Intelligence in Engineering, 11: pp. 143-154.

Salamatov, Y., 1999, "TRIZ: The Right Solution at the Right Time: A Guide to Innovative Problem Solving", Netherlands, Insytec.

Schmidt LC, C. J., 1995, "Guiding conceptual design through behaviour reasoning", Research in Engineering Design 7: pp. 102-25.

Sieger DB, S. R., 1997, "Knowledge representation tool for conceptual development of product designs", Proceedings of the IEEE International conference on Robotics and Automation.

Slimani, K., Ferreira, D.S., Médini, L., Ghodous, P., 2006, "Conflict mitigation in collaborative design", International Journal of Production Research. 44(9): pp. 1681-1702.

Souchkov, V., 1998, "TRIZ: A systematic Approach to Conceptual Design", Proceedings of the Workshop Universal Design Theory, Karlsruhe, Germany.

Sturges RH, O. S. K., Reed R.G., 1993, "A systematic approach to conceptual design." Concurrent Engineering: Research and Applications 1: pp. 93-105.

Sudhakar Y.R., 1996, "Hider: A methodology for early-stage exploration of the design space", In Proceedings of the 1996 ASME Design Engineering Technical Conferences and Computers in Engineering Conference, August 1996. Irvine, California.

Sun, K., Faltings, B., 1994, "Supporting creative mechanical design", In Artificial Intelligence in Design, pages 39-56, 1994. Kluwer Academic Press, Netherlands.

Sycara , K., Navin Chandra, D., Guttal, R., Koning, J., Narasimhan, S., 1991, "CADET: a case-based synthesis tool for engineering design", International Journal of Expert Systems, v.4 n.2, pp.157-188.

Sycara, K., 1990, "Negotiation planning. An AI approach", European Journal of Operational Research, 46, pp. 216–234.

Sycara,K., 1991, "Cooperative negotiation in concurrent engineering design", In D. Sriram, R. Logcher and S. Fakuda (eds), Computer-Aided Cooperative Product Development(Berlin: Springer-Verlag), pp. 269–297.

Takala, T., 1989, "Design transactions and retrospective planning tools for conceptual design", Intelligent CAD systems 2, Springer Verlag.

Tong, C. and Gomory, A., 1993, "A knowledge based computer environment for the conceptual design of small electromechanical appliances", Computers, 26(1), pp. 69-71.

Ulrich, K., Seering, W., 1989, "Synthesis of schematic descriptions in mechanical design", Research in Engineering Design 1: pp. 3-18.

Umeda, Y.I.M., Yoshioka, M., Shimonura, Y., Tomiyama, T., 1996, "Supporting conceptual design based on the function-behaviour-state modeller", Artificial Intelligence for Engineering Design, Analysis and Manufacturing 10(4): pp. 175-88.

Vancza, J., 1999, "Artificial intelligence support in design: a survey", Keynote paper at the 1999 International CIRP Design Seminar, http://www.sztaki.hu/~vancza/papers/AIsurvey1.pdf (Enschede, The Netherlands 1999)

Vescovi, M., 1993, "CFRL: A language for specifying the causal functionality of engineered devices", in: Proceedings Eleventh National Conference on Artificial Intelligence (AAAI-93), Washington, DC.

Vladimir P., 2002, "Laws of Dialectics in Technology Evolution", (Available at http://www.triz-journal.com/archives/2002/06/d/index.htm, last accessed date: 03 April, 2007)

Wang, L., 2002, "Collaborative conceptual design-state of the art and future trends", Computer aided design 34: pp. 981-996.

Wang, Q., R, M., Zhou J., 1994, “Intelligent systems for conceptual design of mechanical products”, New York, Chapman & Hall.

Wei, T., 1977, “An exposition of the I-Ching, or Book of changes”, Hong Kong : Dai Nippon Printing Co.

Werkerman, K. J., 1990, “Multiagent cooperative problem solving thorough negotiation and perspect sharing”, PhD dissertation, Lehigh University.

Wei, C-C., Liu, P-H., C-B Chen, 2000, “An automated system for product specification and design”, Assembly Automation, Vol. 20, No. 3, pp. 225-233(9)

Wong, S. T. C., 1997, “Coping with conflict in cooperative knowledge-based systems”, IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 27, pp. 57–72.

Wynne H., I. M., Y. W., 1998, “Current research in the conceptual design of mechanical products”, Computer aided design 30(5): pp. 377-389.

Xie, J., Stringfellow, A. and Song, X.M., 1998, “Inter-functional conflict, conflict resolution styles, and new product process: a four-culture comparison”, Management Science, 44, S192-0S206.