The Hong Kong Polytechnic University

School of Design

An Agent-Based Framework to Support Collaborative Product Design

By

Jian Xun Wang

A thesis submitted in partial fulfilment
of the requirements for the
Degree of Doctor of Philosophy

May 2007

Certificate of Originality

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

(Signed)

(Name of student) Jian Xun Wang

Abstract

This thesis investigates the requirements of collaborative product design and proposes a new agent-based framework to support collaborative product development by improving the efficiency and effectiveness of the management and coordination of collaborative product design processes.

Collaborative design creates added values in the product development process by bringing the benefits of teamwork and cooperation in a concurrent and coordinated manner. Furthermore, it reduces the loss of efficiency resulting from potential conflicts and misunderstandings among team members. However, the effectiveness and the success of collaborative design are undermined by the difficulties arising from the differences between heterogeneous system architectures and information structures.

Recently, agent technology has been recognised by more and more researchers as a promising approach to analyzing, designing, and implementing industrial distributed systems. The way in which software agents residing in a multi-agent system interact and cooperate with one another to achieve a common goal is similar to the way that human designers collaborate with each other to carry out a product design project. Thus, the hypothesis of this research is that a collaborative product design framework implemented by taking an agent-based approach will be capable of assisting human designers and design teams effectively and efficiently in a collaborative product design process.

Based on the investigation of a collaborative design process, the requirements for a product design framework which functions particularly to support design collaboration are identified. Then, a new framework for collaborative design is proposed and it adopts an agent-based approach and relocates designers, systems, and supporting agents in a unified knowledge representation scheme for collaborative product design.

In order to model the constantly changing design process and the rationales resulting from design collaboration, a Collaborative Product Data Model (CPDM) and a Collaborative Design Process Model (CDPM) are proposed to maintain the consistency of product data and assist designers in making good decisions. The CPDM is established by extending the traditional feature-based

parametric Product Data Model (PDM) to include information related to the management and coordination of a collaborative product design process. It can take maximum advantage of mature technologies and makes the integration of traditional CAX systems easier. The advantage of the CDPM is that, by taking a constraint-oriented approach to modelling the collaborative design process, the collaborative design process can be more efficiently and effectively managed and coordinated because, in product design, most of the design requirements can be represented in computational design constraints. The CDPM helps designers understand the design rationale behind a design decision made by other designers who may be geographically dispersed.

Based on the proposed CPDM and CDPM, the system architecture of an agent-based framework to support collaborative design is proposed, and a prototype system is implemented. This system is capable of assisting designers in the management and coordination of the collaborative design process via the cooperation of a network of agents including mainly management agents, communication agents, and design agents. Design experiments are carried out with the prototype system to validate the feasibility and applicability of the system. The results of these design experiments demonstrate how the proposed framework improves the efficiency and effectiveness of the management and coordination of a collaborative product design process.

This research is located in collaborative product design and field of CSCW (Computer Supported Cooperative Work) and CSCD (Computer Supported Cooperative Design) with an emphasis on evaluating the proposed framework using real design examples. It contributes to the development of new tools supporting collaborative product design using computational agents with unified data structures and software protocols in an integrated framework. The implemented framework with product design examples demonstrated the advantages of using agent technology to achieve better design cooperation, coordination, response, desired modularity, and improved software reusability in the process of product design collaboration.

Acknowledgments

I would like to express my gratitude to everybody who in one way or another helped me throughout the course of this research.

First of all, I would like to express gratitude to my supervisor, Dr. Ming Xi Tang and my cosupervisor Professor John Hamilton Frazer, for their continuous encouragement, and, above all, considerable patience in my development of the thesis.

This Ph.D project was undertaken in the Design Technology Research Centre (DTRC), School of Design, The Hong Kong Polytechnic University. As such, I would like to acknowledge the support from all the staff in the School of Design and all the members in the Design Technology Research Centre. Special thanks go to the fellow PhD. students, Dr. Patrick Janssen, Mr. Kwai Hung Chan, Dr. Jeckie Lee, Dr. Zhen Yu Gu, Miss Tzvetanova Sylvia, Miss Schadewitz Nicole, and my friends, Dr. Shou Qiang Jiang, Dr. Bao Hua Song, Dr. Wei Wang, for their support and friendship.

I would like to acknowledge my family in Beijing, and my parents-in-laws. There are no words to express my gratitude to all of them for their patience, support and encouragement throughout this long journey in pursue of my dreams. Never would I finish this work without their love and absolute support.

Finally, I would like to thank my wife, Lin Nan Song, for living with the thesis as well as me, and for the countless ways she ensured that I finished it in one piece.

Dedication

This thesis is dedicated to my loving mother, Feng Zhen Li.

Publications

Wang Jian Xun and Tang Ming Xi (2007). *Product Data Modelling for Design Collaboration*. In Proceedings of the 11th International Conference on Computer Supported Cooperative Work in Design (CSCWD 2007), Melbourne, Australia, April 26–28, 2007, pp. 321–326.

Wang Jian Xun and Tang Ming Xi (2006). *Design and implementation of an agent-based collaborative product design environment*. Submitted to Computers in Industry.

Wang Jian Xun and Tang Ming Xi (2006). *An Agent-based System Supporting Collaborative Product Design*. Knowledge-Based Intelligent Information and Engineering Systems, Lecture Notes in Computer Science (LNCS), Vol. 4252, pp. 670–677.

Wang Jian Xun and Tang Ming Xi (2006). *An Agent-based Approach to Collaborative Product Design*. In Proceedings of the 2006 ASME International Design Engineering Technical Conferences (DETC) & the Computers and Information in Engineering Conference (CIE), Philadelphia, PA, U.S.A., September 10–13, 2006, ASME DETC2006-99149, pp. 423–429.

Wang Jian Xun and Tang Ming Xi (2006). *A Multi-Agent Framework for Collaborative Product Design*. Agent Computing and Multi-Agent Systems, Lecture Notes in Computer Science (LNCS), Vol. 4088, pp. 514–519.

Wang Jian Xun and Tang Ming Xi (2006). *An Agent-based System Supporting Collaborative Product Design*. Second International Conference on Design Computing and Cognition (DCC'06), Poster Paper, July 10–12, Technical University of Eindhoven, The Netherlands.

Wang Jian Xun and Tang Ming Xi (2005). *Knowledge Representation in an Agent-Based Collaborative Product Design Environment*. In Proceedings of the 9th International Conference on Computer Supported Cooperative Work in Design (CSCWD 2005), Coventry University, United Kingdom, May 24–26, 2005, pp. 423–429.

Wang Jian Xun and Tang Ming Xi (2005). *Development of an Agent-Based Collaborative Product Design Environment*. In Proceedings of the 9th International Conference on Computer-Aided Industrial Design and Conceptual Design (CAID&CD 2005), Delft University of Technology, The Netherlands, May 29–June 1, 2005, pp. 296–301.

Contents

Certificate of Originality I

Abstract II

Acknowledgments IV

Dedication V

Publications VI

Contents VIII

List of Figures XII

List of Tables XV

I Overview 1

1 Introduction 2

1.1 Problem Identification 5

1.1.1 Problems of Computational Design Systems 6

1.1.2 Features of a Collaborative Design Process 7

1.1.3 The Scope of This Research 10

1.2 Research Objectives 11

1.3 Research Methodology 17

1.4 Contributions of This Research 21

1.5 Overview of This Thesis 23

II Research Background 25

2 Research on Collaborative Design 26

2.1 Introduction 26

2.2 Design and Design Process 27

2.2.1 Understanding Design 27
2.2.2 Product Design Process 30
2.2.3 Design For X 33
2.3 Computer Enabling Technologies for Design 34
2.3.1 Computer Supported Cooperative Work (CSCW) 35
2.3.2 Intelligent Agents and Multi-Agent Systems 39
2.4 Product Data Modelling 44
2.4.1 Research on Product Data Modelling 45
2.4.2 Limitations 47
2.5 Product Design Process Modelling 48
2.5.1 Research on Product Design Process Modelling 50
2.5.2 Limitations 52
2.6 Collaborative Product Design Techniques 54
2.6.1 Web-based Paradigm 56
2.6.2 Client-Server Paradigm 59
2.6.3 Agent-based Paradigm 61
2.6.4 Evaluations 64
2.7 Summary 66

III Research Proposition 68

3 Towards An Agent-based Collaborative Design Framework 69

3.1 Introduction 69
3.2 Design Management and Coordination 70
3.3 An Agent-based System Framework 75
3.4 Design Agent 79
3.5 Knowledge Representation 82
3.5.1 CPDM 84
3.5.2 CDPM 85
3.6 A Collaborative Design Example 87
3.7 Summary 89

4 Collaborative Product Data Modelling 90

4.1 Introduction 90
4.2 The Requirements 91
4.3 CPDM – A Collaborative Product Data Model 93
4.3.1 Overview 93
4.3.2 Collaborative Design Management Data Module 95

4.3.3 Product Design Coordination Data Module 98
4.3.4 Product Data Module 101
4.4 An Example of CPDM 103
4.5 Summary 106

5 Collaborative Product Design Process Modelling 108

5.1 Introduction 108
5.2 The Motivations 109
5.3 CDPM – A Collaborative Design Process Model 110
5.3.1 Overview 110
5.3.2 Formal Definition 112
5.3.3 A Framework of Design Collaboration 115
5.4 Summary 118

6 Implementation Strategy of the Prototype System 120

6.1 Overview 120
6.2 The Primary Programming Language 122
6.3 The Agent Development Framework 124
6.4 Internal Structure of the Software Agents 127
6.5 Integration Issues with Traditional CAD Platform 129
6.6 Summary 132

7 Validation and Discussion 134

7.1 Introduction 134
7.2 The Design Example 134
7.2.1 Starting the System 135
7.2.2 Project Planning and Scheduling 137
7.2.3 Project Definition and Design Specifications 142
7.2.4 Design Process Cooperating and Coordinating 145
7.2.5 Collaborative Design Process Tracking 153
7.2.6 Final Design Result 154
7.3 Evaluation and Discussions 155
7.4 Summary 156

VI Conclusions 157

8 Conclusions and Future Work 158

8.1 Research Conclusions 158
8.2 Future Work 162

8.2.1 Extension of the Collaborative Product Data Model 162
8.2.2 Extension of the Collaborative Design Process Model..... 163
8.2.3 Further Development of the Project Management Agent 164
8.2.4 Development of a Product Design Ontology 164
8.2.5 More Experiments..... 165
8.2.6 Consideration on Human Factors..... 165

Bibliography **167**

List of Figures

Figure 1.1: The system development research process (Adapted from the model of research process of systems development research methodology proposed by Nunamaker, et al. [1991].) 20

Figure 2.1: Typical life cycle of a product (Diagram is redrawn from [Pahl and Beitz, 1996].).....29

Figure 2.2: French's model of the design process.....31

Figure 2.3: Pahl and Beitz's model of the design process.....33

Figure 2.4: Research and development contexts for Computer Supported Cooperative Work (CSCW). Each ring represents a work level (community, organisation, project/team, small group, or individual) and its corresponding systems (listed directly below the hub of the figure) and research areas (left) (Adapted from Grudin's model of the areas of related research within CSCW [Grudin, 1994]). 37

Figure 3.1: The proposed computational model of collaborative product design management and coordination.....72

Figure 3.2: The system architecture and hardware configuration of the proposed agent-based framework supporting collaborative product design.....76

Figure 3.3: The main components of the proposed design agent.....80

Figure 3.4: The main data modules of the proposed Collaborative Product Data Model.....84

Figure 4.1: The proposed Collaborative Product Data Model (CPDM) 94

Figure 4.2: The product assembly tree, design variables, design constraints, and the relationship among them 101

Figure 4.3: The design of a dining table with a chair.....103

Figure 5.1: The function requirement (FR) tree 112

Figure 5.2 The mapping process among the function requirements and the design variables.....114

Figure 5.3: The constraint-based Collaborative Design Process Model (CDPM).....116

Figure 6.1: The hardware environment for the development of the prototype system 121

Figure 6.2: The JADE architecture.....126

Figure 6.3: Internal structure of the software agents.....127

Figure 6.4: The integration of Autodesk Inventor with the design agent and the JADE platform.....131

Figure 6.5: A simplified case of Autodesk Inventor's object hierarchy.....132

Figure 7.1: The main graphical user interface of the Agent Manager (JADE agent run-time environment) in which the Project Management Agent, the Product Data Management Agent, the Design Coordination Agent have already been created 136

Figure 7.2: The main graphical user interface for Project Management Agent 138

Figure 7.3: Specify the properties of a task using the task properties dialog box.....139

Figure 7.4: Define dependency relationships between tasks using the task properties dialog box.....139

Figure 7.5: The resource list and the resource usage information.....140

Figure 7.6: Assign resources to a task using the task Properties dialog box.....141

Figure 7.7: An email sent to Designer1 (Jason Wong) stating the task assignment generated by the Project Management Agent 141

Figure 7.8: Key anthropometric dimensions required for the design of a set of dining table and chairs (The data is taken from Human dimensions of Chinese adults, GB 10000-88, A 95th percentile height would indicate that only 5% of the study population would have heights of greater dimension. A 5th percentile height would indicate that 95% of the study population would have heights of greater dimension.).....143

Figure 7.9: Main graphical interfaces of the Design Communication Agent....144

Figure 7.10: The Design Agent panel is used to specify the collaborative design management data.....145

Figure 7.11: Using the Design Agent to specify the collaborative design Coordination data.....146

Figure 7.12: Drawing of a dining table with selected design parameters 147

Figure 7.13: Design parameters for the dining table 148

Figure 7.14: Drawing of a chair with selected design parameters 149
Figure 7.15: Design parameters for the chair 149
Figure 7.16: Defining a design constraint on the assembly/sub-assembly level .150
Figure 7.17: A list of design constraints defined for the dining table and chair assembly.....151
Figure 7.18: The message showing a design constraint is violated and its reason 152
Figure 7.19: The updated design project progress.....153
Figure 7.20: Final design of a set of dining table and chairs.....154

Figure 8.1: Various design constraint views in the form of matrix.....163

List of Tables

Table 2.1: A simple classification of CSCW systems..... 38

Table 2.2: A summary of research works on web-based collaborative design.... 59

Table 2.3: A summary of some typical collaborative design systems based on the traditional client-server paradigm 60

Table 2.4: A summary of some typical research works for agent-based collaborative design 63

Table 4.1: The detailed data structure of the Collaborative Design Management Data module..... 96

Table 4.2: The detailed data structure of the Design Coordination Data module..... 99

Table 4.3: The detailed data structures of the ConstraintItem and the CoordinationItem 100

Table 4.4: The detailed product data of the collaborative design management data module of a dining table instance, and a chair instance 104

Table 4.5: The detailed product data of the design coordination data module of a dining table instance, and a chair instance..... 105

Table 4.6: ConstraintArray1 (Constraints in the design coordination data module for the dining table instance)..... 105

Table 4.7: CoordinationArray1 (CoordinationHistory in the design coordination data module for the dining table instance)..... 105

Table 7.1: Four types of dependency relationships between tasks 140

Part I

Overview

Part one consists of an introduction chapter that gives an overview of this thesis. First, the research background is introduced and key problems of a collaborative product design process are identified. The main research objectives are then defined and an outline is given of the research proposition. Finally, methodological issues are discussed.

Chapter

1

Introduction

Design is in essence a collaborative process. In the context of developing computer-based design systems to support such a process, previous research and development work in this area did not address the issues of design support agents operating in a collaborative environment until recently, i.e. the beginning of 90s and the recent years of the widespread Internet use.

Collaboration is a structured, recursive process where two or more people work together to achieve a common goal – typically an intellectual endeavour that is creative in nature – by sharing knowledge, learning and building consensus.

In the field of design, increasing competition resulting from economic globalisation and rapidly changing customer requirements are forcing major alterations of the ways in which products are designed, manufactured, and maintained among many organisations. The benefits of collaboration are evident since collaborative design can help manufacturers raise product quality, reduce development costs and shorten the lead time. Collaborative design creates added

values in the design and production process by bringing the benefits of teamwork and cooperation in a concurrent and coordinated manner. Furthermore, it reduces the loss of efficiency resulting from potential conflicts and misunderstandings among team members.

However, all parties involved in a design project such as marketing, designing, manufacturing, assembling, testing, quality controlling and even suppliers and customers are often geographically dispersed. They usually work with different and sophisticated tools and software systems that require consistent maintenance and management as a result of frequent upgrading or changing in the software.

Furthermore, design data and knowledge are usually stored in various different formats with information structures and coding methods for distribution over the network and heterogeneous systems. All these tend to undermine the effectiveness and success of collaborative design among multidisciplinary designers.

Multi-agent systems are being considered by many as one of the most promising technological paradigms for the development of open, distributed, cooperative, and intelligent software systems [Hao, et al., 2006]. In such systems, software components or systems are designed in such a way that they can act like agents who have self-contained information and knowledge to complete tasks. An intelligent agent itself consists of self-contained knowledge-based sub-systems capable of perceiving, reasoning, adapting, learning, cooperating, and delegating in a dynamic networked environment to tackle specialist problems. Such intelligent software agents can interact with one another within a collaborative framework to assist human designers or design managers in carrying out product design projects, instead of isolated individual design tasks, which often have to be integrated with additional resources and attention separately.

The use of agents in design has been explored by many researchers in the field of design computing and cognition. However, this research is still at its early stage of development and most of the currently developed prototypes of such agent-based design systems are being tested on abstract examples. No commercial applications or generic frameworks of agent-based design support systems are available for demonstrating how agent technology can contribute to design collaborations.

The design and implementation of an agent-based collaborative product design framework capable of assisting designers in the management and coordination of the collaborative design process is the main focus of this thesis. In this research, a new multi-agent framework that relocates designers, managers, systems, and supporting agents as well as integrating traditional CAD software in a unified knowledge representation schema for product design is developed. In order to model the constantly evolving design process and capture the rationales resulting from design collaboration, a Collaborative Product Data Model (CPDM), which extends traditional Product Data Model (PDM), and a constraint-based Collaborative Design Process Model (CDPM) are integrated to form the basis of the agent-based collaborative product design framework. A prototype system of this framework is implemented and evaluated using real design examples.

In this thesis, a number of key elements that constitute an agent-based framework are identified and the main challenges to implement such a framework that support collaborative design are tackled. The thesis provides experimental and application results for a good understanding of the relevance of agent technology to design collaboration.

1.1 Problem Identification

Product design is a complex problem solving and knowledge refining process in which information and knowledge of diverse sources are processed simultaneously by a team of designers involved in the life phases of a product [Tang, 1997].

Nowadays, products are becoming more and more complex than ever in terms of structure, material, technology, etc. The quantity and diversity of information that designers must deal with can become tremendous. It is not uncommon that hundreds of teams of designers with different background collaborate with each other in a big complex design project. Thus, product design becomes a social activity which relies on cooperation across multidisciplinary designers or design teams involved in the product life-cycle, each with their own backgrounds and thus different perspectives on the design. Therefore, a successful collaborative product design has a far heavier reliance on communication and cooperation. From another perspective, product design is becoming a knowledge intensive process in which sharing and reusing various kinds of knowledge involved in the product life-cycle is crucial. So, a new product design environment should be a knowledge based system which allows the generation and flexible exchange of design knowledge and information to create more added values to the product being designed.

From a process point of view, product design is a highly organised process in which the design is created and refined incrementally. Traditional product design models employ a sequential process that breaks the design task into subtasks that are executed serially in a predefined style in a centralised hierarchical organisation. Such design models are now considered by many to be insufficiently flexible because of ever-changing customer requirements and the dynamic and

distributed nature of product configuration and manufacturing process. In such models, the lack of information feedback from low-level manufacturing activities to the high-level design, numerous iterations, and few design alternatives, are obvious. As a result, this leads to inefficient, expensive and time-consuming designs.

Currently, popular commercial CAX (CAD/CAM/CAE) software, e.g. AutoCAD®, Autodesk Inventor®, SolidWorks®, Pro/Engineer®, CATIA®, and etc, already provide user-friendly graphic interfaces, 3D solid modelling, and many kinds of physical simulation functionalities in product design. However, in these systems there is limited support provided for the management and coordination of the collaborative design process.

1.1.1 Problems of Computational Design Systems

An emerging new field of Computer Supported Collaborative Work (CSCW) provides opportunities for developing better collaborative design systems. The difficulties in supporting design collaboration are related to consistency maintenance of design constraints and detecting conflicts throughout the whole product design process. In the past two decades, accompanying the advancement of computer and internet technologies, many efforts have been taken to study the enabling technologies and tools to support the communication and cooperation using CSCW techniques. Infrastructure level enabling techniques, such as synchronous/asynchronous communication tools, data sharing, distributed database system, concurrency control, coordination control, and etc, have been tackled frequently by academia and industry. In the meantime, in the field of design computing, many efforts have been undertaken to develop enabling techniques suitable for solving design problems in an engineering context. Gero et al. [1993] and Liu [1993] worked on shape emergence in the design process.

Rosenman et al. [1996] introduced a model for multidisciplinary design collaboration in which multiple views of an object are implemented based on functional context. Liu et al. [2002] presented a multi-agent cooperative design environment supporting evolutionary design via the cooperation of a group of agents. Wang [1999] introduced a unique combination of machining-feature, agent technology, and function block to tackle and facilitate concurrent design problems with due consideration of downstream constraints under a distributed environment.

However, relatively few research activities have been conducted to establish a computational foundation for a collaborative product design framework in which design support functionalities are provided by intelligent and collaborative software agents.

1.1.2 Features of a Collaborative Design Process

The investigation into the development of large complex products reveals that a large scale design process is characterised by the following features:

- People working on the same product design project often appear to be temporally and spatially distributed. Thus, quite a lot of time and money are spent for planning, tracking, and coordinating the whole product design process.

- Collaborative design processes usually last for a long time. Due to the evolving customers' preferences, improving technologies, and the shifting macroeconomic environment, the requirements imposed on the product may need to be changed during the product development process.

- During the product design process, a lot of decisions are taken by each participant involved from different perspectives to meet various functional criteria. These decisions influence each other and interdepend on each other. Whenever a decision is poorly justified or badly communicated, the whole design process suffers setbacks and delays.

- The sharing and reusing of various kinds of design knowledge involved in the product life-cycle is very common, but misunderstanding occurs when such information is presented in an unfamiliar fashion for some of the design team members.

- The hardware and software systems for collaborative design are often heterogeneous (they may have different system architectures or information structures). The collaboration among the designers using these systems becomes difficult as a result of losing information in the format conversion process.

From the features examined above, certain requirements for a collaborative product design framework with design management and coordination functionalities can be identified:

- Distributed design project management tools should be integrated to provide support to project managers in the design project management work, such as creating design projects, decomposing design tasks, planning, scheduling, and implementing the design process, allocating and coordinating interdependent design tasks, as well as managing all the resources involved, including people, software tools, and hardware tools.

- Effective and systematic mechanisms for design coordination and constraint propagation are required and designers should be automatically

notified of any design constraint conflict as well as any design change being made by others.

- Design tasks should be managed in a way in which everyone involved in the design process understands the rationale behind key design decisions.

- Multi-modal communication tools, such as message, video conference, whiteboard, application sharing, etc., should be provided to facilitate the design knowledge sharing and design work cooperation among distributed designers.

- The framework should support the easy integration of heterogeneous software and hardware tools used by distributed designers, providing easy access to product data and design knowledge, as well as convenient means to exchange design information among the designers.

- Popular CAD software should be integrated and Human Computer Interaction (HCI) should be user-friendly, so that the extra burden imposed on designers to get familiar with the collaborative design framework can be kept as low as possible.

- Product data management tools are needed to accelerate product development cycles by facilitating product data reuse and sharing, and protecting it from inadvertent changes.

As mentioned above, intelligent agents are autonomous, self-organised and thus very suitable for dealing with distributed problems. The way in which intelligent software agents residing in a multi-agent system interact and cooperate with one another to achieve a common goal is similar to the way that human designers collaborate with each other to carry out a product design project. Thus, it is

believed that a collaborative product design framework implemented by taking an agent-based approach will be capable of assisting human designers or design teams effectively and efficiently in product design and help address some of the above issues.

1.1.3 The Scope of This Research

The scope and context of this research is the development of a collaborative product design framework employing an agent-based approach. As such, the following problems are addressed in this thesis:

- What are the possible ways in which design collaboration can be facilitated in an agent-based collaborative product design framework?
- What differences are there between traditional design process and the network-supported distributed collaborative design process and how to coordinate the collaborative design process and resolve the design conflicts?
- What project control strategies should be adopted to determine an appropriate project decomposing structure, delegate subtasks to individual design agents, develop and refine plans and schedules for achieving the global design goal, while supporting both constraint propagation and concurrency where possible, so that a product design project can be managed efficiently and effectively while the overall product design process progresses in a timely and accurate fashion?
- How the existing heterogeneous CAD/CAM/CAE/PDM software tools on diverse platforms can be integrated seamlessly or improved to suit such a collaborative product design framework?

- What agents should be included to build an agent-based collaborative product design framework? How to model the overall system structure, the internal structure of an individual agent and the communication mechanisms between agents?

- What design knowledge should be represented in the agent-based framework and in what means this knowledge can be represented and managed properly?

- What are the human-computer interaction (HCI) issues to be solved such that the proposed framework will not add a major burden to users and at the same time users are capable of accessing their favourite tools conveniently?

The development of computational agents with unified data structures and software protocols contributes to the establishment of a new way of working in collaborative design, which is increasingly becoming an international practice. The abovementioned problems motivated the research in this thesis and the design and implementation of an agent-based collaborative product design framework which is capable of assisting designers in the management and coordination of a collaborative design process. Establishing a computational framework supporting collaborative product design in which some of the abovementioned problems can be addressed with some real design examples to validate the applicability of the framework is the focus of this research.

1.2 Research Objectives

This research is located in collaborative product design and field of CSCW (Computer Supported Cooperative Work) and CSCD (Computer Supported

Cooperative Design) with an emphasis on evaluating the proposed agent-based framework using real design examples with certain complexities. The overall goal of this research is to design, implement, and validate a collaborative product design framework which adopts an agent-based approach and is capable of facilitating collaborative design across the geographical barriers and supporting the management and coordination of the collaborative design process. On the one hand, this agent-based framework aims to support the whole design process in terms of design project management, design communication, and sharing of product data and design tools. On the other hand, it aims to support both the embodiment design stage and the detail design stage further in terms of design process coordination.

To achieve the goal set out above, this research is conducted to include the following study and experiments:

- To review the state of the art research in the development of collaborative product design systems.

- To identify the requirements and points of view related to the emerging technologies for supporting collaborative product design through a field study.

- To design the system architecture and identify the key component agents to tackle two of the most essential problems in collaborative design, i.e. the management and coordination of the collaborative design process and the development of software agents that support such a process.

With these main objectives, the following computational issues are researched and experimented.

- To develop a collaborative product data model to support design knowledge management in a distributed framework.
- To develop a collaborative product design process model to facilitate better design collaboration.
- To design and implement a prototype of the proposed agent-based collaborative product design framework.
- To validate the proposed agent-based collaborative product design framework though real design examples.

These research objectives are pursued in the following theoretic and practical context of design research, from the perspectives of Artificial Intelligence and computational design.

(1) To review the state of the art research in the development of collaborative product design systems.

In order to identify the problems and challenging research issues in the field of developing collaborative product design systems, a state of the art review and investigation of the research work in relevant areas is carried out. These relevant research areas include theories and methodologies of product design and collaborative work, approaches for product data modelling and design process modelling, advances of the development of collaborative design systems, agent technologies, etc.

(2) To identify the requirements and points of view related to the emerging technologies to support collaborative product design through a field study.

In order to develop a collaborative product design framework that really suits the cooperation needs of designers who are often geographically and temporally dispersed, it is very important to identify the function requirements for such a framework. Then, based on the investigations on the emerging technologies that may be used to develop the framework, the requirements and points of view related to these emerging technologies to support collaborative product design are identified.

(3) To design the system architecture and identify the key component agents to tackle two of the most essential problems in collaborative design, i.e. the management and coordination of the collaborative design process and the development of software agents that support such a process.

Based on the requirements identified for a novel collaborative product design framework, the system architecture of the proposed agent-based collaborative product design framework is designed. The proposed agent-based framework is represented as a network of intelligent agents which interact with each other and with the participating team members to facilitate collaborative design work in an open environment. It integrates not only design knowledge but also other resources involved such as those heterogeneous hardware and software tools.

(4) To develop a collaborative product data model to support design knowledge management in a distributed framework.

In order to support design knowledge management, a collaborative product data model that incorporates both the information related to the management and coordination of the collaborative design process and three dimensional geometrical modelling data of the product is developed. In this research, this model is established by extending traditional popular parametric feature-based

product model, including two additional data modules: Collaborative Design Management Data, and Design Coordination Data.

(5) To develop a collaborative product design process model to facilitate better design collaboration.

In order to facilitate better design collaboration, a collaborative product design process model which can identify, capture and represent design knowledge of the constant evolving collaborative design process is developed. In this research, a constraint-based approach is adopted to formulate the collaborative design process. A key issue in developing such a model is how to coordinate the design process effectively and efficiently and, at the same time, keep the design process in a timely and accurate fashion without constraining it too much which is supposed to be free and dynamic in the first place. A constraint-based coordination strategy is employed to augment the capabilities of individual designers, as well as enhance the ability of collaborators to interact with each other with the support of computational resources as software agents.

(6) To design and implement a prototype of the proposed agent-based collaborative product design framework.

Based on the proposed system structure and the proposed design knowledge representation schema, i.e. the collaborative product data model and the collaborative design process model, a prototype of the agent-based collaborative product design framework is designed and implemented. Enabling technologies are studied to improve the extensibility of the framework and make it architecture-neutral, network-centric. In the prototype, management agents, design agents, and tool agents are developed. Design and engineering tools/services including product database, design knowledge base, share/private

workspace, and video conference system, are also integrated to interact with human designers in an open environment.

(7) To validate the proposed agent-based collaborative product design framework though real design examples.

In order to validate the applicability of the proposed agent-based collaborative product design framework, real design examples are selected to illustrate how human designers can be supported in the collaborative design process management and coordination using the developed prototype system. Furthermore, limitations of the framework are identified to suggest directions for future improvements.

Meanwhile, the relevance of agent technology to design collaboration is also examined. This provides an insight on how mature agent technology can be used to support collaborative design in a realistic scale, and how designers' thinking might be affected by the introduction of such new technologies in the near future. This is necessary since in the past two decades, many technologies, such as Internet/Web, agents, client/server based on CORBA, Java, MS COM/DCOM, were employed to implement distributed collaborative design systems. One of the objectives of this research is to identify and address the key technical problems in implementing a prototype of the proposed agent-based framework and to show how agent technology can improve design collaboration with the implementation of product design management agents, design agents, and tool agents in a distributed framework.

Since the framework developed in this research aims at providing intelligent support to designers in a distributed collaborative design process, a designer-centred approach is taken in this research. All the above mentioned objectives are

centred at the overall goal of this research which is to design and implement a collaborative product design framework to make the cooperation among geographically and temporal dispersed designers more effective and efficient.

1.3 Research Methodology

The objective of the research has been defined as the development of an agent-based collaborative product design framework to facilitate the management and coordination of a collaborative product design process in which geographically distributed designers are involved in a common complex product design project.

In order to achieve the research objective and address the problems which have been analysed in Section 1.1, this research follows a paradigm which is adapted from the model of systems development research methodology proposed by Nunamaker, et al. [1991]. Despite the fact that the original research methodology was developed for the research domain of Information Systems (IS) research[1] and the proposed agent-based collaborative product design framework in this research might be considered outside the scope of what is commonly considered to be an Information System, it is in fact very suitable to this research because from perspective of a software agent, processing information or knowledge is the main way in which it acts in a computational framework.

As shown in Figure 1.1, the research process consists of three key stages which include five research steps totally.

Stage I: Conceptualization. During this stage, research problems are identified and the framework is conceptualized. It includes the design of the system

[1] A common definition of the objective of Information Systems (IS) research is "the effective design, delivery, use, and impact of information technologies in organizations and society" [Keen, 1987].

architecture and the development of methods, and the theoretical models that will be employed in the framework. This stage consists of three research steps:

The construction of a conceptual framework. The development of an intelligent collaborative product design framework is justified to be of great interest to product design companies and manufacturers. The system requirements and research problems in the context of designing and implementing such a framework are examined carefully. The system requirements are stated so that they are measurable and thus can be validated at the system evaluation stage. A clear definition of the research problem provides a focus for the research throughout the development process. At the same time, a thorough and in-depth state-of-the-art review in several fields related to the current research activities is conducted to discover additional novel approaches and ideas which could be incorporated in the proposed framework. Those fields include those relevant to the collaborative product design and those relevant to the enabling technology.

The development of a system architecture. In this step, a system architecture of the proposed agent-based collaborative product design framework is developed. The system architecture puts the system components that compose the agent-based framework into the correct perspective, in which the functionalities are specified, the structure relationships and dynamic interactions among those system components are defined. The system architecture provides a roadmap for the building process of the agent-based collaborative product design framework.

The analysis and design of the system. In this step, alternative approaches to implement the proposed framework is examined and evaluated. The specifications of the framework are developed to be used as a blueprint for the implementation of the framework. Data structures and databases are designed. Especially, a design knowledge representation schema including the Collaborative Product Data

Model (CPDM) and the constraint-based Collaborative Design Process Model (CDPM) is developed.

Stage II: Implementation. During this stage, the prototype of the framework is implemented. This stage consists of one research step:

The building of a prototype of the system. In this step, enabling technologies are examined and the choice of appropriate programming language and programming environment to develop the framework is made. Then, a prototype of the proposed agent-based collaborative product design framework is implemented. The implemented prototype system will be used to demonstrate the feasibility of the design and the usability of the functionalities. Furthermore, the process of implementing a prototype system can provide insights for others on what complex issues are involved in the computational aspects of developing such a framework.

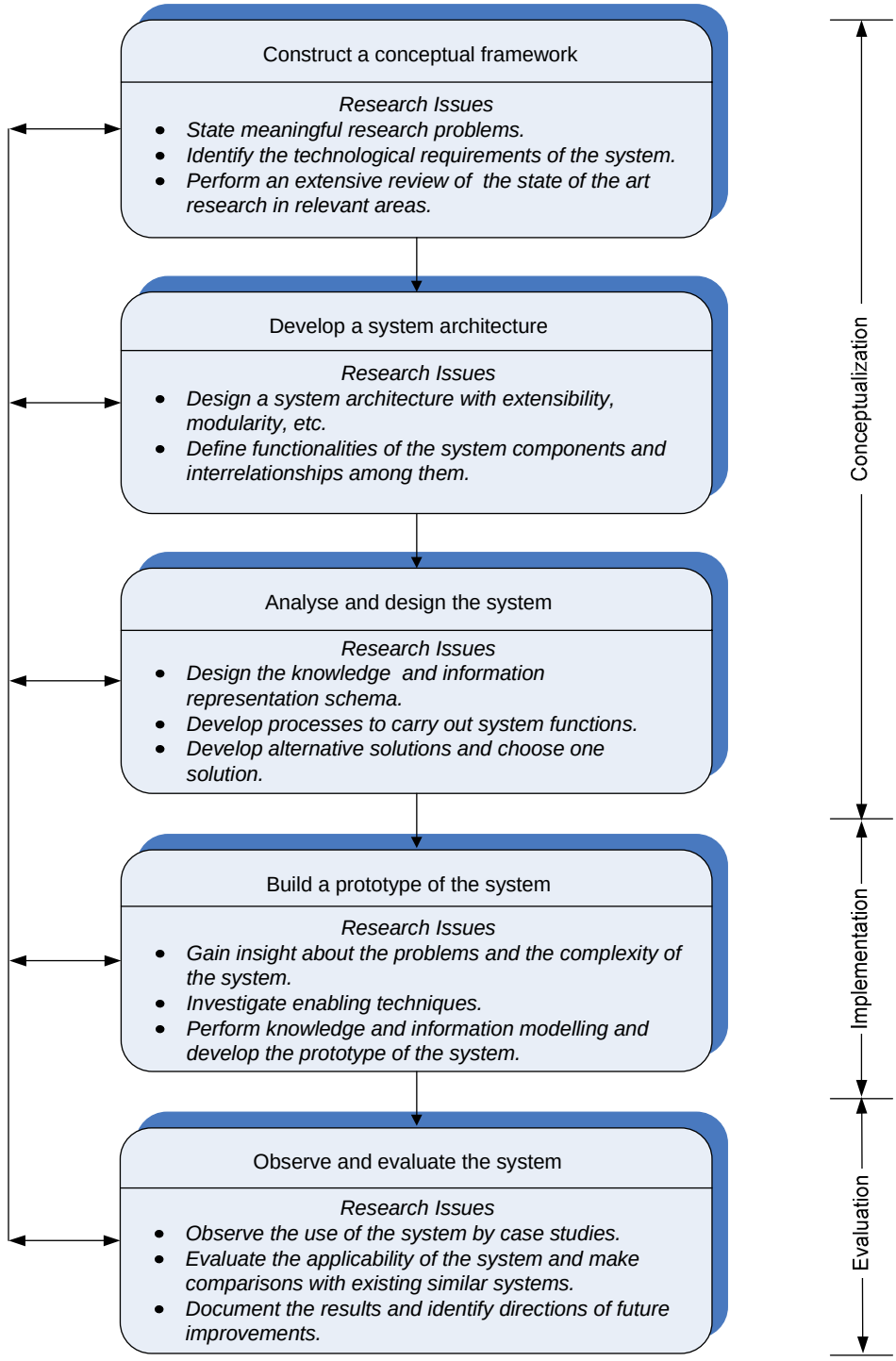


Figure 1.1: The system development research process (Adapted from the model of research process of systems development research methodology proposed by Nunamaker, et al. [1991].)

Stage III: Evaluation. During this stage, the prototype system of the framework is evaluated to test its applicability. This stage consists of one research step:

The observation and evaluation of the system. In this step, the performance and usability of the proposed framework is examined and its impacts on individuals, groups, and organizations are observed. Real design examples are used to demonstrate how the proposed agent-based collaborative product design framework improves the efficiency and effectiveness of the management and coordination of a collaborative design process.

1.4 Contributions of This Research

In the previous discussion of problems arising from the design and development of large complex products, the importance and benefits of successful collaborative product design have been highlighted. Many critical issues must be addressed in order to develop a collaborative product design framework.

The major contributions of the research are as follows:

- This research establishes the relevance of agent technology and collaborative design and demonstrates the advantages of using agent technology in the collaborative product design process in a distributed multi-user environment, thus providing a theoretical foundation for the development of an internet-based collaborative product design environment for initiating collaborative design projects.

- By adopting an agent paradigm, a loosely coupled system architecture incorporating a network of intelligent agents is established to support collaborative design. Any newly developed agent providing certain type of

service can join the framework as long as it complies with the FIPA[2] specifications. Furthermore, this architecture can provide a higher-level interoperability among heterogeneous systems and software.

- A Collaborative Product Data Model (CPDM) is developed to overcome the weakness of traditional popular feature-based parametric product data models. In the proposed CPDM, information related to the collaborative product design management and coordination is more effectively incorporated. At the same time, through the extension of the traditional popular feature-based parametric product data model, it can take the advantages maximally of the mature technologies and make the integration of traditional CAX systems that usually employ a feature-based parametric product data model.

- A constraint-based Collaborative Design Process Model (CDPM) is developed. Design constraints are in essence the computational representation of the product design requirements. This research takes a constraint-oriented approach to modelling the collaborative design process. In this way, the collaborative design process can be more efficiently and effectively managed and coordinated by software agents. In addition, it helps designers understand each other better when design conflict occurs.

- A type of design agent encapsulated traditional CAD software, Autodesk Inventor®, as well as integrating design knowledge base is proposed to support designers in the collaborative design process.

[2] FIPA: The Foundation for Intelligent Physical Agents (FIPA) is a body for developing and setting computer software standards for heterogeneous and interacting agents and agent-based systems. Its official website is http://www.fipa.org.

To achieve these contributions, based on the proposed system architecture, employing the proposed CPDM and CDPM to represent the collaborative design knowledge, a proof-of-concept prototype of the agent-based collaborative product design framework that integrates multiple design tools, such as design software, communication tools, distributed database, etc., has been developed. With the prototype system developed in this research, it is possible to study how distributed designers involved in a design project can cooperate with each other more effectively and efficiently, with the demonstration of a real design example.

1.5 Overview of This Thesis

This thesis is divided into four parts.

Part one consists of one chapter. It gives an overview of the research including the research background, the research problems, the research objectives, and the research methodology.

Part two consists of one chapter. It reviews the state of the art work related to this research. Relevant research fields include the product design and design process, computer enabling technologies, research in computer aided product design, and research in multi-agent collaborative product design systems, etc.

Part three presents the main proposition made by this thesis, and consists of five chapters. Chapter 3 gives an overview of the system structure of an agent-based collaborative product design framework. Chapter 4 describes the proposed Collaborative Product Data Model (CPDM) and Chapter 5 presents the proposed constraint-based Collaborative Design Process Model (CDPM). Chapter 6 gives the technological details of the implementation of a prototype system. In Chapter 7, real collaborative design examples are used to demonstrate the applicability of

the proposed framework. The design examples are used to evaluate the framework and identify the main advantages of using agent technology for collaborative design and in what degree such a technology can benefit designers and in what way the approach still has limitations or barricades for its full scale applications in a wide range of design domains.

Part four consists of one chapter. It concludes this research and lists several issues for further research in this area.

Part II

Research Background

Part two consists of one chapter. It reviews the state of the art work related to this research. Relevant research fields include product design and design process, computer enabling technologies, research in computer aided product design, and research in multi-agent collaborative product design systems, etc.

Chapter

2

Research on Collaborative Design

2.1 Introduction

The research work presented in this thesis attempts to design and implement a collaborative product design framework by adopting an agent-based approach, thus it builds on foundational work in relevant areas of research including product design and design process, design representation methods, computer enabling technologies, computer supported collaborative design systems, etc.

In this chapter, a state-of-the-art review of the work in the relevant areas of research is carried out. It compares the differences between these researches with the proposed work. Further references to literature with regard to more detailed problems encountered during the development of the prototype system will be made in later chapters when these problems are discussed in detail.

2.2 Design and Design Process

2.2.1 Understanding Design

Design covers a broad spectrum of activities, such as mechanical design, engineering design, fashion design, architecture design, graphics design, typography design, etc. Thus, the definition of design is widely interpreted and argued. Different definitions of the concept of design are given by many different researchers and organisations.

Design is defined by Caldecote [1986] as: "The process of converting an idea into information from which a new product can be made". Matchett [1968] stated that "Design is to find the optimum solution to the sum of the true needs of a particular set of circumstances". The International Council of Societies of Industrial Design (ICSID)[3] [2004] defines design as: "Design is a creative activity whose aim is to establish the multi-faceted qualities of objects, processes, services and their systems in whole life-cycles". The Accreditation Board for Engineering and Technology (ABET) [1985] defines design as: "Engineering design is the process of devising a system, component, or process to meet desired needs. It is a decision-making process (often iterative), in which the basic sciences and mathematics and engineering sciences are applied to convert resources optimally to meet a stated objective. Among the fundamental elements of the design process are the establishment of objectives and criteria, synthesis, analysis, construction, testing, and evaluation".

In this research, design is studied and discussed in the context of product design practice and designer is used synonymously for design and development engineer. The word "product" concerned in this research can be mechanical product, consumer product, and whatever requires three-

[3] ICSID: International Council of Societies of Industrial Design. Its official website is http://www.icsid.org/.

dimensional solid modelling techniques with computer-aided design software for the creation of their concepts, dimensions, and configurations. The definition of design proposed by Dym [2004] can be usefully borrowed for this research and is adapted here to give a definition of product design:

Product design is a systematic problem-solving process and it is concerned with the development of detailed specifications for artefacts (products) whose form and function achieve stated objectives and satisfy specified constraints.

- *Artefacts (products)* are human-made objects, i.e. the products which are being designed. They are most often in a physical form, like ships, automobiles, carburettors, tables.

- *Form* is the shape, i.e. the geometry of the artefact being designed.

- *Function* means those things that the artefact is supposed to do.

- *Specifications* of artefacts are precise descriptions of the properties of the product being designed. They are typically numerical values of performance parameters (i.e. constants or variables that serve as indicators of the artefact's behaviour) or of attributes (i.e. properties or characteristics of the artefact).

From a systematic perspective, product design is considered an essential part of the product life cycle [Pahl and Beitz, 1996]. It involves various activities which aim at finding solutions to technical problems related to the whole product life cycle. As shown in Figure 2.1, the activity of design is triggered by a market need or a new idea and starts with product planning and ends, when the product's useful life is over, with recycling or environmentally safe disposal. As such, it requires multiple expertises from various different domains to apply their scientific and engineering knowledge to address technical problems and determine the properties of the product in terms of

function, safety, ergonomics, production, transport, operation, maintenance, disposal and recycling. In general, the objectives of a product design are defined in terms of specific requirements which can be called *function requirements* (FRs). Then, to satisfy these function requirements, a physical embodiment characterised in terms of *design parameters* (DPs), i.e. the detailed specifications of the artefact, must be created [Suh, 1990].

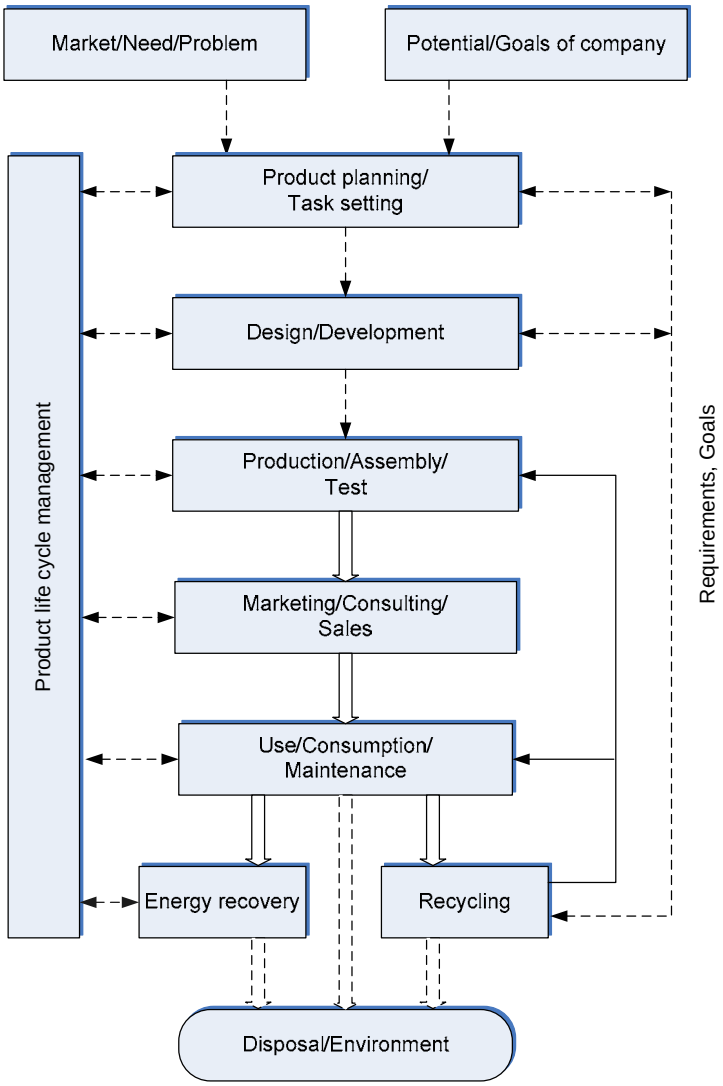


Figure 2.1: Typical life cycle of a product (Diagram is redrawn from [Pahl and Beitz, 1996].)

The product design process begins with the establishment of FRs in the functional domain to satisfy a given set of customer needs, and ends with the

creation of the artefact with DPs that satisfy these FRs. The specified FRs provide a basis for evaluating the proposed designs, that is, the FRs provide the "targets" against which the success of achieving the design objectives can be measured.

2.2.2 Product Design Process

As stated in the definition given in the previous section, product design is a systematic problem-solving process. Hurst [1999] stated in his book (*Engineering Design Principles*) that good design is critical for success both in national and export markets and could only be ensured by adherence to a formal design process. Furthermore, a design process can support designers by functioning as a framework or methodology.

There have been many attempts in developing models of the design process and all these models can be categorised into three types:

- Descriptive models,
- Prescriptive models, and
- Computer-based models.

Descriptive models describe the sequences of activities that typically occur in designing while prescriptive models prescribe a better or more appropriate pattern of design activities. A computer-based model, in contrast, describes a method that can be represented in computer programs that may assist human designers in accomplishing a design task. In this section, only the first two types of design process models are reviewed. A detailed review on computer-based models of design process will be given in Section 2.5. In this research, a constraint-based Collaborative Product Design Process Model, proposed to facilitate the management and coordination of a collaborative product design process, will be described in detail in Chapter 5.

2.2.2.1 Descriptive Models

Models of this type can be found in [French, 1985; Cross, 1997; Popovic, 2003]. They reflect the solution-focused nature of design thinking. They usually generate a concept design early which is subjected to analysis, evaluation, and if any fundamental flaw is found, then the design is refined and developed further. This cycle goes on until a satisfactory solution is found. Such a process is heuristic: using previous experience, general guidelines and rules of thumb that lead to what the designer hopes to be the right direction, but with no absolute guarantee of success [Cross, 2000].

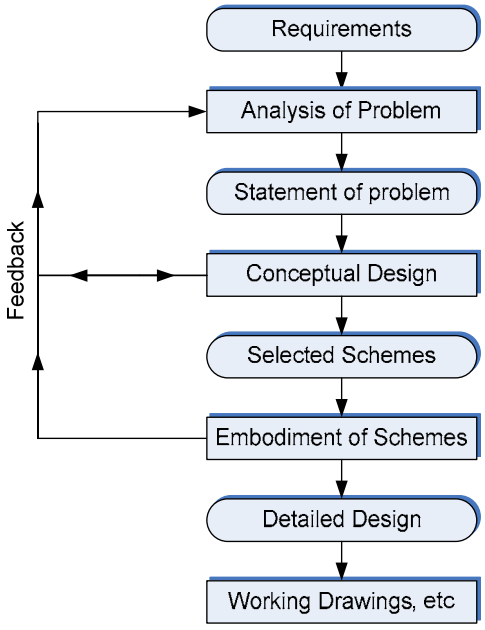


Figure 2.2: French's model of the design process

Figure 2.2 shows the descriptive model of the design process proposed by French [1985]. In the diagram, the stages reached, or outputs are represented by rectangles with round corners and the activities, or work in progress are represented by rectangles with straight corners. The design process is triggered by a need, and the first design activity is to analyse the problem to develop the statement, i.e. design specifications. The phase of conceptual design takes the statement of the problem and generates broad solutions to it in the form of schemes. These schemes are worked out in greater detail and, if there is more

than one, a final choice is made in the phase of embodiment of schemes. The detailed design is the last phase. There is usually a lot of feedback from the phase of embodiment design to the conceptual design phase.

2.2.2.2 Prescriptive Models

Models of this type can be found in [Archer, 1984; Pahl and Beitz, 1984; March, 1984; Jones, 1984; VDI, 1987]. Compared with descriptive models of design process which simply describe a more-or-less conventional, heuristic process of design, prescriptive models are concerned with attempting to persuade or encourage designers to adopt improved ways of working and usually offer a more algorithmic, systematic procedure to follow [Cross, 2000]. Prescriptive models put much emphasis on the analytical work prior to the generation of several alternative design concepts among which the best would be selected based on a rational design decision.

One of the most referenced models of this type is the design process model proposed by Pahl and Beitz [1996]. As shown in Figure 2.3, their model consists of four design phases which are outlined below:

- Clarification of the task: analyse the design problems, identify the requirements and design constraints, and formulate a product proposal.
- Conceptual design: establish function structures, search for suitable solution principles, and combine those principles into a working structure.
- Embodiment design: starting from a concept, designers determine the layout and forms and develop a product in line with the technical and economic criteria.
- Detail design: the arrangement, forms, dimensions and surface properties of all the individual parts are finally laid down, materials

specified, technical and economic feasibility re-checked, and all the drawings and other production documents produced.

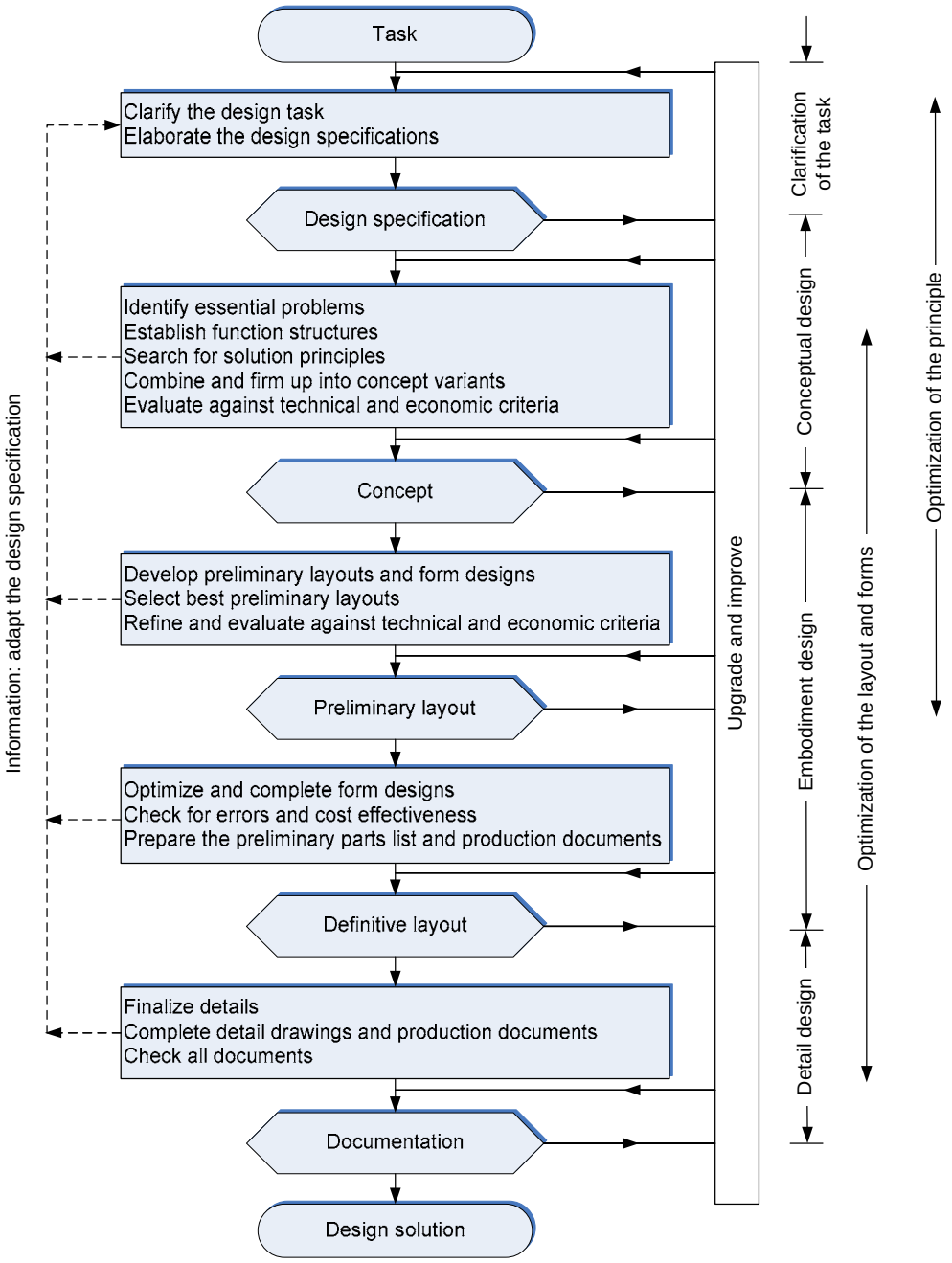


Figure 2.3: Pahl and Beitz's model of the design process

2.2.3 Design For X

It is a commonly accepted view that the design of a commercial product is a compromise among conflicting design goals. For example, on the one hand,

the performance of a product is expected to be maximised, on the other hand, the development time and cost is expected to be minimised. In efforts to improve the efficiency of product design and maintain cost-effective benefits, some approaches to incorporating knowledge of the various downstream phases of the product life-cycle have been developed. The most commonly used of these approaches is "Design for X (DFX)" in which X can be interpreted as either a life-cycle process, such as manufacture or assembly, or a design quality criteria, such as cost, aesthetics, ergonomics, serviceability or a diverse collection of these processes or design properties [Benhabib, 2003; Boothroyd, 1996, 2002; Bralla, 1996].

The need for the DFX methodology is identified as design engineers become increasingly aware of a lack of appropriate detailed knowledge in important product life-cycle processes. The indication of DFX for computational design technology is that supporting design activities must go far beyond mere support for geometric modelling and constraint management and a novel product design framework must be capable of supporting cooperation and interaction between members of a multidisciplinary product design team.

2.3 Computer Enabling Technologies for Design

Over the past few decades, many research activities have been carried out in order to improve product development efficiency by employing computing techniques throughout the design and development process. Alteration of the ways in which products are traditionally made with the conventional configuration of manufacturing organisations to meet the challenges resulting from economic globalisation and rapidly changing customer requirements can improve the product development efficiency, increase product life-cycle revenue and reduce development costs.

In Hong Kong, for example, the success of transition to a knowledge based economy is largely dependent on creativity and innovation led by design, taking advantage of Pearl River Delta area in mainland China being a world manufacturing centre. However, the need for changing the design tools and environments which are still more or less based on single users to team based collaborative environments has just been realised, but the research in this area is still limited.

Because of the complex nature of design and technology, it is now nearly impossible for an individual to tackle the design and development of a new complicated product design project single-handed and most of the design tasks can no longer be modelled in a single user computer program. Increasingly, qualitative design knowledge, incomplete information, ever-changing customer requirements demand computational support for more effective and efficient collaboration and management in a distributed framework. These all have given rise to the development of approaches that support multidisciplinary decision making and design cooperation in a concurrent engineering context with integrated applications of Computer Supported Cooperative Work (CSCW). In these developments, two computer enabling technologies, namely, Computer Supported Cooperative Work (CSCW) and intelligent agents, have emerged with greater potentials than that promised by Artificial Intelligence several decades ago.

2.3.1 Computer Supported Cooperative Work (CSCW)

In engineering design and manufacturing, the use of networked data by sharing and communication through sophisticated computer systems for the development, documentation, exchange, and modification of product data has become a common approach. Those single user based systems and software for design support are being or will be replaced soon by software systems and environments with increased intelligent support dealing with knowledge

sharing, and facilitating cooperation and learning. Computer Supported Cooperative Work (CSCW) is an emerging area for addressing the design and deployment of computer-based technologies in support of cooperation, collaboration, communication and coordination among members of groups, teams, organisations who may be geographically dispersed [Bannon, 1992; Greif, 1988; Grudin, 1994; Ye, 2003]. Since 1996, there has been an annual international conference, CSCW in Design (CSCWD)[4], dedicated to the research, development, and application of CSCW technologies to the design of complex artefacts and systems.

In Figure 2.4, the history of CSCW and its development and research contexts can be illustrated by Grudin's model which depicts the areas of related research within the domain of CSCW [Grudin, 1994]. Since the invention of computer technology, various research works in related areas have contributed to the development of CSCW. These research works include advances in Management Information System (MIS) and Information Technologies (IT) since 1965, the work in Software Engineering (SE) and Office Automation (OA) since the mid 1970's, and the development of the Internet in the 1990s.

Grudin's model is helpful for considering not only the history of CSCW research, but also for the broadness of the applications of CSCW technologies. Therefore, problems resolved and lessons learnt from early MIS and OA technologies remain valuable for researchers in the design of workflow technologies, communication tools and information sharing mechanisms.

[4] CSCWD: The International Working Group on Computer Supported Cooperative Work in Design (CSCW in Design), its official website is http://www.cscwd.org.

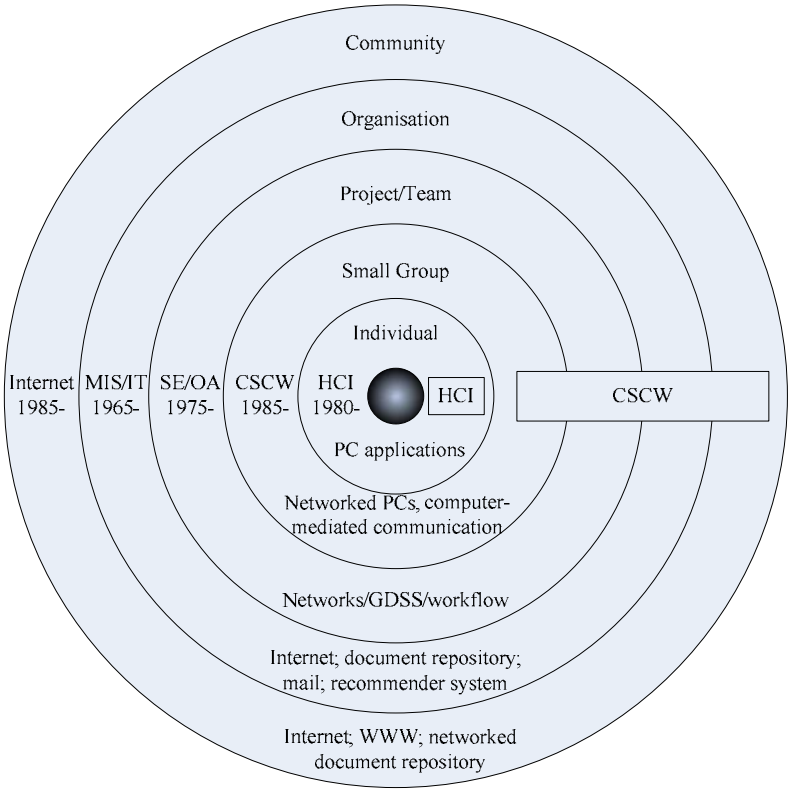


Figure 2.4: Research and development contexts for Computer Supported Cooperative Work (CSCW). Each ring represents a work level (community, organisation, project/team, small group, or individual) and its corresponding systems (listed directly below the hub of the figure) and research areas (left) (Adapted from Grudin's model of the areas of related research within CSCW [Grudin, 1994]).

MIS: Management Information Systems; IT: Information Technologies;
SE: Software Engineering; HCI: Human Computer Interaction;
GDSS: Group Decision Support System OA: Office Automation;
WWW: World Wide Web

Consideration of the social processes that underpin cooperative and collaborative activities is driven by the shift from designing for the single user interacting with a computer interface to designing for collaboration and coordination among multiple users using different computer terminals. Lyytinen [1989], in his report "*Computer Supported Cooperative Work: Issues and Challenges*", suggested:

CSCW is neither solely a tool or a technology business and not just a new way to study computer impact on the workplace. Instead, in CSCW, equal emphasis is put on the distinctive qualities of co-operative work processes, and on questions of design: how to mould computer technology to fit into and support these work processes.

Therefore, CSCW research is based on the comprehensive understanding of the nature of cooperative and collaborative work arrangements, and people's existing work practices and interpersonal relationships. Based on this understanding, the technologies to facilitate and maintain those relationships are then developed.

In general, there are two dimensions, time and place, that make up the CSCW domain as shown in Table 2.1.

Table 2.1: A simple classification of CSCW systems

	Same Time	**Different Times**
Same Place	Face-to-face meeting (meeting room, office)	Asynchronous interaction (project scheduling, coordination tools)
Different Places	Synchronous distributed interaction (shared editors, video conferences)	Asynchronous distributed interaction (email, instant messenger, bulletin board, and other message systems)

Difficulties arise in carrying out collaborative work when cooperative tasks cannot be successfully carried out because of time differences and distances. Lack of instantaneous access to necessary information can bring about bottlenecks in work processes. Process matters also arise due to lack of awareness of the cultural differences in others' work practices and in their working conditions.

So far, a number of technologies and concepts have been developed to satisfy CSCW goals. Technologies like audio and video conferencing, e.g. Microsoft

NetMeeting®, support information sharing and instant communication. Virtual workspaces and shared document repositories allow data to be shared through intranet and internet, thus support synchronous and asynchronous communication and facilitate the fulfilment of complicated co-design and co-creation projects [Gutwin and Greenberg, 1996]. Group calendars help coordinate users' schedules and meeting support systems provide assistance in structure content sharing and decision making [Kincaid et al., 1985 and Palen, 1999]. Workflow systems are useful for aiding teams/groups in coordinating their activities when work procedures are well defined [Hales and Lavery, 1991]. Finally, with the prevalence of wireless technologies, people are able to keep communications and access work materials while on the move [Brown et al., 2001; Churchill and Wakeford, 2001]. The state-of-the-art technologies supporting CSCW can be found in [McCARTHY, 1994 and Qiang, 2002].

2.3.2 Intelligent Agents and Multi-Agent Systems

Over the past few years, multi-agent systems have been perceived as a crucial technology not only for effectively exploiting the increasing availability of diverse, heterogeneous, and distributed on-line information sources, but also as a framework for building large, complex, and robust distributed information processing systems which exploit the efficiencies of organized behaviour [Lesser, 1999].

2.3.2.1 Intelligent Agents

Similar to the case for the definition of design, so far, there is no unified agreement on the definition of intelligent agent yet. According to the Agent Platform Special Interest Group [2000] of the Object Management Group (OMG)[5], it is described as: "An agent can be a person, a machine, a piece of

[5] Object Management Group (OMG): It is an international, open membership, not-for-profit computer industry consortium. OMG Task Forces develop enterprise integration standards for a wide range of technologies. OMG's modelling standards enable powerful visual design,

software, or a variety of other things." For Wooldridge [2002], "An agent is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives."

Nwana [1996] has given a classification of software agents which identified seven types of agents: collaborative agents, interface agents, mobile agents, information/internet agents, reactive agents, hybrid agents, and smart agents. A detailed introduction for intelligent agents can be found in the book of Wooldridge [2002].

There is usually some confusion as to understanding agents and objects[6]. In fact, there are at least three distinctions between them. First, agents are more autonomous than objects, and in particular, they decide for themselves whether or not to perform an action on request from other agents; second, agents are capable of performing flexible (reactive, pro-active, social) behaviour, while the standard object model has nothing to say about this type of behaviour; third, a multi-agent system is inherently multi-threaded, in that each agent is assumed to have its own thread of control, while in the standard object model, there is only a single thread of control in the system.

To satisfy the design objectives, there are four basic components that constitute a typical agent:

(1) An explicit description of the problem domain, such as communication protocols and the ontology framework;

(2) Functional codes to carry out a given task;

execution and maintenance of software and other processes. Its official website is http://www.omg.org/ .

[6] Object: In computer science, an object is an individual unit of run-time data storage that is used as the basic building block of programs. Each object is capable of receiving messages, processing data, and sending messages to other objects. Each object can be viewed as an independent little machine or actor with a distinct role or responsibility.

(3) Inter-agent communication mechanisms (including languages, protocols and translation modules);

(4) A user-interface that is customised for the task domain.

2.3.2.2 Multi-agent Systems

When two or more agents interact or work together to carry out a set of tasks or achieve a set of goals, they may form a multi-agent system (MAS). In this thesis, the terms 'agent-based system' and 'multi-agent system' are used synonymously. Characteristically, one individual agent in the multi-agent system will not have all the data or all the methods available to achieve an objective and thus will have to collaborate with other agents. Also, these agents may be homogeneous or heterogeneous. As with distributed agents, data is decentralized and execution is asynchronous. The earlier research field in multi-agent systems is Distributed Artificial Intelligence (DAI) which focuses on the development of solutions and computational models for representing, analysing and implementing the patterns of interaction and coordination for large complex distributed problems.

In general, there are two models of agent-based systems in terms of the organisation of multiple agents.

One is the well-known blackboard architecture which was proposed originally in the HEARSAY project as a means to organise and control large AI systems [Erman, et al., 1975]. With the blackboard architecture, all information about the problem is stored in a global common data repository and is shared by all the agents in the system. This model is used in some projects in engineering design, such as DICE [Sriram, et al., 1992], DESIGN-KIT [Stephanopoulos, et al., 1987], CONDOR [Iffenecker, 1994], EXPORT [Monceyron, et al., 1992].

For the second model of agent-based system, each agent is independent and has its own representation of the situation independent from that of other

agents. They build their own models of the current solution by acquiring information from other agents. This type of system needs a protocol and a format for the messages for expressing requests and replies. Each agent stores the current solution, or at least part of the solution, in its local database. Some projects such as MARS [Abriat, 1991], PACT [Cutkosky, et al., 1993], First-Link [Park, et al., 1994] and Next-Link [Petrie, et al., 1994], DIDE [Shen, et al., 1995] have used this model of communication.

The blackboard architecture suffers several shortcomings while it has the obvious advantage of ensuring consistency. There are three main reasons identified by Talukdar [1992]. First, it is based on a centralised knowledge base which is difficult to expand. Second, it does not allow parallel processing of modules that interface with the blackboard. Third, the blackboard system imposes a supervisory control and coordination of activities. In the engineering design domain, each design tool may have its own representation of the data describing the problem to solve and such a representation is often private for the tool. So, there is no particular advantage in transferring it to a centralised location. Most importantly, during a complex design project lifetime, new tools may be introduced, changing the structure of the design environment. New independent agents can be easily, to a certain degree, incorporated into the agent-based system implemented with the second model. Furthermore, the characteristic that there is no global control on agents in the system is more suitable for implementing an open, distributed system. Based on above analyses, the second model is employed in this research project for developing a software architecture of an agent-based design framework.

One of distinctive properties of the agent architecture proposed in this research project is decentralisation, i.e. of perspectives, control and operations. Through autonomy of individual agents, the framework seeks to eliminate the need for supervisory control over individual activities. Moreover, addition, modification, or elimination of a single agent within the framework should not

require reconfiguration of other agents. These characteristics allow the system architecture to be open and scale to large, more complex systems gracefully [Gruber, et al., 1992]. The suitability of such an approach for supporting design related activities will be discussed later in Chapter 3 and Chapter 7.

Lesser [1999] argues that multi-agent systems provide a powerful model for computing in the 21st century, in which networks of interacting, real-time, intelligent agents seamlessly integrate the work of people and machines, and dynamically adapt their problem solving to effectively deal with changing usage patterns, resource configurations and available sources of expertise and information. Parunak [1998] has analysed where agent technology can be best used in design operation activities and he stated that “agents are best suited for applications that are modular, decentralized, changeable, ill-structured, and complex.”

An intelligent agent consists of self-contained knowledge-based systems capable of perceiving, reasoning, adapting, learning, cooperating, and delegating in a dynamic environment to tackle specialist problems. The way in which intelligent software agents residing in a multi-agent system interact and cooperate with one another to achieve a common goal is similar to the way that human designers collaborate with each other to carry out a product design project. Thus, the hypothesis of this research project is that a collaborative product design framework implemented by taking an agent-based approach will be capable of assisting human designers or design teams effectively and efficiently in a collaborative product design process. Furthermore, the following advantages are expected to be achieved: better system openness, desired modularity, flexible simulation, more reliability, better responsiveness and improved software reusability.

2.4 Product Data Modelling

Product data modelling is one of the key techniques for the development of computer-aided design systems. It is concerned with the definition, representation, manipulation, and management of complex product data elements as well as various relationships among them. During the design process, the product data is constantly evolving and the properties of the product data elements as well as the relationships among them are always changing. Therefore, a computer-aided product design environment should not only support the representation of the end product and the relationships between its components, but also the evolution of the components and associated knowledge over the whole life cycle of the product [Shen and Barthes, 1995].

In the preface for a special issue of the journal *Computers in Industry* that was dedicated to papers on reporting the research of product data and product design process modelling, Pels [1996] gave two reasons for the development of techniques for product data modelling. The first is to aid the designer in imagining and validating the new product development. The second reason is to communicate the product development process. It is even more the case while the product to be designed becomes more complicated and multidisciplinary designers are involved in the product creation and development process with the assistance of a collaborative product design environment.

Popular CAX systems, e.g. AutoCAD®, Autodesk Inventor®, SolidWorks®, Pro/Engineer®, CATIA®, etc. employ feature-based parametric models which mainly include data information related to geometry structure and manufacturing features. Due to lack of information about design management and coordination, it is obvious that such a feature-based parametric model is not rich enough to ensure effective and efficient collaborative design process

management and coordination, and the sharing of product data in a distributed design environment.

2.4.1 Research on Product Data Modelling

So far, there has been a lot of attention to the issue of product data modelling for computer-aided design. To summarise, most of the researches in this area fall into the following three categories.

- The first is the development of a neutral product data model which aims at facilitating product data exchange between different CAX systems [ISO, 1991a, 1991b, 1991c, 1992a, 1992b, 1992c, 1992d, 1994a, 1994b; NIST, 1996; Gorti, et al., 1998].

- The second is the development of product data models that embody design rationale and knowledge which intend to promote the share and reuse of product data [Gero, 1990; Gero, et al., 1992; Alberts, 1994; Simoff and Maher, 1998; Wong and Sriram, 1993a, 1993b; Rosenman and Wang, 1999, 2001].

- The third is the development of product data models for a particular design domain or with special purposes [Gu and Chan, 1995; Shen and Barthes, 1995; Zha and Du, 2002].

Neutral product data models in the first category focus on facilitating product data exchange between different CAX systems. The typical one in this category is the STEP standard, i.e. the international Standard for the Exchange of Product Model Data, officially ISO 10303 [ISO, 1991a, 1991b, 1991c, 1992a, 1992b, 1992c, 1992d, 1994a, 1994b] developed by the International Standards Organization (ISO). Its objective is to provide a mechanism that is capable of describing product data throughout the life cycle of a product, independent from any particular system. STEP is able to model geometry,

manufacturing features in machining domain, and some non-geometry information such as specifications (e.g. tolerance, surface finish), assembly relations, etc. so that it can provide a basis for flexible exchange of product model data. Another typical product data model in this category is the Initial Graphics Exchange Specification (IGES) published by NIST[7] which also aims to allow the digital exchange of information among different CAD systems. Using IGES, a CAD user can exchange product data models in the form of circuit diagrams, wireframe, and freeform surface or solid modelling representations. Most of the other product data models in this category were developed based on the STEP and tried to better represent the design object with richer semantics than just geometry information. Gorti et al. [1998] proposed an object-oriented representation scheme for product data which was geared to accommodate current and future STEP-based standards.

Product data models in the second category aim to promote the share and reuse of product data by embodying design rationale and knowledge. A typical one is the paradigm of '*purpose-function-behaviour-structure*' proposed by Gero and Rosenman in order to provide a multiple view of a product to facilitate the multidisciplinary collaborative design [Gero, 1990; Gero, et al., 1992; Rosenman and Gero, 1994, 1996; Rosenman and Wang, 1999, 2001]. In their model, the *structure* is what an artefact is; the *behaviour* is exhibited as a result of a certain structure under given conditions and results in certain *function*s being performed; the *purpose* defines those functions which are intentional and defines why an artefact is and what it does or what it is for. Wong and Sriram (1993a, 1993b) have also developed a similar product data model employing an object-oriented representation approach. Some other models in this category use ontology to represent the design knowledge and

[7] NITS: The National Institute of Standards and Technology, formerly known as The National Bureau of Standards, is a non-regulatory agency of the United States Department of Commerce's Technology Administration.

incorporate it into the product data model [Alberts, 1994; Simoff and Maher, 1998].

Product data models in the third category are usually developed for a particular design domain or with special purposes. Shen and Barthes [1996] proposed an object-oriented product data model supporting the representation of the components of mechanical products and of relationships among the components and developed an object-oriented programming environment MOSS for implementing its features. Zha and Du [2002] developed a STEP-based model for concurrent integrated design and assembly planning of mechanical systems by incorporating entities of integrated resources into the STEP-based model. Gu and Chan [1995] developed a STEP-based generic product modelling (GPM) system to integrate a variety of manufacturing activities in a concurrent engineering environment. Wang and Nnaji [2004] developed a constraint-enabled distributed product data model capable of incorporating not only static relations among design objects, but also dynamic relations/constraints to facilitate design collaboration.

2.4.2 Limitations

Product data modelling plays a vital role to enable product data sharing, communicating design rationales behind the design decision, as well as coordinating the design process. However, current product data models are not sufficient for collaborative design. They suffer from a number of limitations:

- For the first category, only information of geometry structure, assembly relations, and some manufacturing specifications are included in the product data models. They lack information about distributed design coordination and design rationales which are important for design collaboration.

- For the second category, design rationales are considered and represented in the product data models such that a multiple view can be provided to a multidisciplinary design team. However, they lack design coordination information which is required to coordinate a distributed collaborative design process.

- For the third category, although some product data models are proposed to support product data and design knowledge sharing [Zha and Du, 2002; Wang and Nnaji, 2004], the same problems occur with them as with those in the second category and they lack design coordination information.

- Another limitation with those product data models proposed to support collaborative design is that most of them are defined in a thoroughly new way and this results in a great deal of redundant information. Furthermore, they have not taken full advantage of the mature feature-based parametric product data models. Therefore, it is difficult to integrate them with currently popular CAX systems.

To deal with the limitations found with the current product data models, a Collaborative Product Data Model (CPDM) is proposed by extending traditional feature-based parametric Product Data Model (PDM) to include information related to the management and coordination of a collaborative product design process. This will be described in detail in Chapter 4.

2.5 Product Design Process Modelling

In Section 2.2.2, both descriptive and prescriptive models of the design process which attempt to provide a top level view of the human cognitive design process with good visibility of the design objectives are discussed. However, they are so abstract that it is nearly impossible to express them with

computer codes for complex multi-parameter product design problems. In fact, they are often used as a reference in order to improve the existing product design processes. In this section, computer-based approaches to product design process modelling are reviewed.

A large complex product design project carried out in a collaborative design environment often involves various multidisciplinary people and teams who must coordinate their activities based on information flow, available resources, and other constraints. Moreover, because design is a non-deterministic process, it is often necessary to capture incomplete information that evolves over time. A formal and comprehensive method for modelling computer-aided collaborative design process is of many interests. Park et al. argue [1999] that there are at least three important motivations for seeking such a method as follows.

(1) Design process automation

While collaborative design processes are likely to benefit from computer support systems, they are too complex and extend over too wide a range of functions to apply local automation without incurring side effects. A formal description of process is needed for (a) addressing various problems arising from missed or misunderstood dependencies between process components, (b) selecting and designing local design process automation (implementation of computer agents), (c) understanding and coordinating the collaborative design work, and (d) defining a conflict management strategy.

(2) Reduction of design process cycle-time

Global market competition has driven product manufacturers to reduce the time-to-market cycle time through Business Process Reengineering (BPR) and other means. Increasingly, design processes that previously were conducted

serially are now performed simultaneously, thus putting greater demand on effective and efficient communication and coordination.

(3) Coordination of distributed activities

Advancements in computer network technology offer new ways for design engineers and other specialists to work over the network. Product design activities are increasingly being carried out using computer tools in a distributed fashion at all stages of product development. This has increased both the need for and difficulty of generating, maintaining and sharing a common understanding of events among collaborators [Cutkosky, et al., 1996].

2.5.1 Research on Product Design Process Modelling

Modelling a product design process involves capturing, representing and evaluating various events whose internal interactions are not always evident [Park and Cutkosky, 1999]. However, capturing a complicated product design process over the lifetime of a project in a consistent and unambiguous manner is a challenging problem. Also, modelling a product design process at multiple levels of granularity is even more difficult. This is especially the case for a design process in which multidisciplinary designers and engineers are involved in a distributed collaborative product design environment.

Over the years, there have been a number of methods proposed to model the computer-aided product design process. To summarise, most of them view a product design process as a number of design tasks and describe it by representing the relationships among these tasks. These models can be classified into the graph-based approach such as digraph, PERT/CPM, SADT/IDEF0, and the matrix-based approach such as DSM. In these models, a task is defined at an appropriate level with required inputs and outputs generated and it may also be decomposed further into a number of sub-tasks. All the tasks and their associated sub-tasks form a design task tree. Following

is a review of these methods partly based on previously published comparisons [Gebala and Eppinger, 1991; Park, 1995; Khoo, et al., 2003].

(1) The graph-based approach

Digraph: A directed graph (or digraph) is the simplest means to represent a design process with nodes representing tasks and directed edges representing the required information flow between tasks. It provides an intuitive view of a process, and is quite effective when the numbers of nodes and edges are small.

PERT/CPM: The Program Evaluation and Review Technique (PERT) which was invented by United States Department of Defense's US Navy Special Projects Office in 1958 is a very popular model. It is often associated with the Critical Path Method (CPM) and aims to analyse the tasks involved in completing a given project, especially the time needed to complete each task, and identifying the minimum time needed to complete the total project [Meredith, et al., 1985; Wiest and Levy, 1977].

SADT/IDEF0: The Structured Analysis and Design Technique (SADT) is a dataflow-oriented design approach and was originally developed by Ross [1977]. SADT provides a mechanism for representing hierarchical abstraction of tasks and supports a graphical notation for indicating the tasks and their associated information inputs and outputs. The IDEF0[8] representation was developed based on SADT and has been used to represent a large and complex design process in some recent research work [Qureshi, et al., 1997].

(2) The matrix-based approach

[8] IDEF0: The term "IDEF" stands for ICAM DEFinition language which is the standard for modelling and analysis in management and business improvement efforts. IDEF is a product of the Integrated Computer-Aided Manufacturing (ICAM) initiative of the United States Air Force. It covers a range of uses from function modelling to information, simulation, object-oriented analysis and design and knowledge acquisition. Specifically, IDEF0 is a functional modelling language building on SADT, and IDEF1X addresses information models.

DSM: The Design Structure Matrix (DSM) was originally suggested by Steward [1981a, 1981b]. The DSM employs an adjacency matrix representation to overcome the size limitations of preceding graph-based process models and provides an elegant way to detect and manipulate iteration loops. In a DSM, each task is assigned to a row and a corresponding column of the matrix. The information dependencies between two tasks are marked in the corresponding cells of the matrix. The use of adjacency matrix can result in a compact representation of the design process. Furthermore, through matrix operations, the DSM provides a mechanism to optimise a design process by "partitioning" and "tearing" the design structure matrix [Gebala, et al., 1991].

To address process modelling issues in large-scale complex design projects, Park and Cutkosky [1999] have also proposed an engineering process representation and modelling tool named as Design Roadmap (DR). The DR framework was proposed to be a superset of the aforementioned graph-based process models and matrix-based models (such as PERT, IDEF0 and DSM) and it is capable of representing design cycles, design task abstraction hierarchy, and constraint relationships between design tasks. It has been used to develop and document engineering processes, to construct functional diagrams of integrated systems, to create process templates, and to manage collaborative projects. Mori, et al. [1999] have also developed a similar method for modelling and planning the design process.

2.5.2 Limitations

The effective modelling of product design process plays an important role in the success of the development of a product. However, current product design process models are not sufficient for the development of a computer-supported collaborative design system and suffer from a number of limitations:

- For the graph-based approach, process models including digraph, PERT/CPM and SADT/IDEF0, have a common disadvantage that they are weak and ineffective to capture large and complex design processes. Because design processes tend to be fairly complex with the increase in the size of design processes, this makes the tracing of design cycles extremely difficult and even impossible.

- As for the matrix-based approach, the DSM method has its own limitations. First, the dependency relationships between connected nodes are not immediately apparent and this makes redundant and parallel paths difficult to distinguish. Second, it assumes a well-defined task decomposition exists in advance and does not facilitate model expansion and refinement.

- A common characteristic of the prior design process models is that they attempt to facilitate project task planning and management by focusing on the dependencies pertaining to the information flow between tasks. However, the design constraint, as an important element of product design, is not addressed directly by these process models.

A design constraint, in its broadest definition, is a limitation on the permissible values of a design element (i.e. feature, form, or component) within itself or with respect to other design elements. In this research, a constraint-based Collaborative Design Process Model (CDPM) is proposed and the constraints are formulated as a complex, dynamic, and hierarchical constraint network. In CDPM, a collaborative design process is viewed as a sequence of state transitions from one design state to another resulting from the design decisions made by participating designers and the precondition of design state transition is the consistency of all design constraints being preserved. There are a number of advantages by taking a constraint-oriented approach to modelling the collaborative design process. The collaborative design process can be more

efficiently and effectively managed and coordinated because, in product design, most of the design requirements can be represented in computational design constraints. Furthermore, the CDPM helps designers understand the design rationale behind a design decision made by other designers who may be geographically dispersed. In Chapter 5, the proposed CDPM is presented in detail and its advantages are discussed.

2.6 Collaborative Product Design Techniques

Designers have long used computers for their calculations and the development of Computer Aided Design (CAD) has gone through several major stages in the past. The first stage can be traced back to the development of interactive computer graphics and the introduction of the CAD term in the 1960s. During the 1970s, wireframe and surface modelling software became available and the importance of CAD was realised gradually by government, industry, and academia. During the 1980s, the main application thrust of CAD came with the availability of personal computers and workstations. Many CAD theories and algorithms were developed and solid modelling software became available. Since then, with computers becoming more and more affordable, CAD implementations have evolved dramatically and its application areas have gradually expanded in the 1990s. To meet the increasing demand for worldwide design collaboration and outsourcing trends in product manufacturing, the most recent research and development issues are aimed at renovating the CAD systems in order to make them support distributed design collaborations.

In a collaborative product design system, multidisciplinary distributed designers and engineers can share their work and cooperate with each other more efficiently and effectively via the internet/intranet. So far, there have been extensive research works carried out aiming at developing methodologies

and prototype systems to facilitate distributed design collaborations. Also, CAD software vendors have been trying to seize the huge business opportunities in this area and have developed some types of collaborative design tools or integrated them into their traditional standalone CAD systems. According to the main technologies and the system structure employed to implement the collaborative design functionalities, there exist three categories of collaborative design systems, namely, web-based collaborative design systems, collaborative design systems based on the traditional client-server paradigm, and agent-based collaborative design systems.

- *Web-based collaborative design systems* are developed by mainly adopting the World Wide Web as a collaboration platform with the web server working as a repository of design information. Such a collaborative design system primarily aims to provide designers with functionalities such as design data sharing, product design review and mark-up, tracking of product file version change, etc.

- *Collaborative design systems based on the traditional client-server paradigm* refer to those implemented by mainly employing a traditional client-server approach, such as Microsoft distributed COM (DCOM), Common Object Request Broker Architecture (CORBA), Java family of technologies including Remote Method Invocation (RMI), JINI, Enterprise JavaBeans, etc. Research projects of this type usually aim to provide distributed designers with more collaboration support functionalities, such as design project management, design process coordination, and etc.

- *Agent-based collaborative design systems* are developed by mainly employing the emerging agent technologies. Research projects in this field attempt to take full advantage of agent technologies and usually

attempt to provide design collaboration functionalities similar to those provided by those employing a traditional client-server paradigm.

In this section, the related research works and commercial CAD systems are surveyed according to the above three categories. In each category, their implemented functionalities, supported collaborative design strategies, system architectures, and enabling information technologies are examined. This review is partly based on previously published comparisons [Li, et al., 2005; Wang, et al., 2002].

2.6.1 Web-based Paradigm

The World Wide Web ("WWW" or simply the "Web") is the most far-reaching and extensive medium of information exchange among the available technologies. It provides a lightweight, easy-to-use, and operating system independent platform for users to search, browse, retrieve, and manipulate information dispersed around the world easily at their desktops.

The ability of the web for designers to combine multimedia to publish information relevant to the spectrum of the design process, from concept generation and prototyping to virtual manufacturing and product realisation, motivated the adoption of the web as a collaborative product development tool [Shen and Wang, 2003]. The web is now increasingly playing vital roles in developing collaborative product design systems. Its use primarily aims to provide multidisciplinary distributed designers with functionalities such as design data sharing, product design review and mark-up, tracking of product file version change, etc. These functionalities are generally realised through the implementation of some kinds of application services running on the web server side which users can access and manipulate via a web browser or add-on viewers in some CAD systems running on the client desktop side.

During the past decade, a number of collaborative product design systems have been developed by adopting a web-based infrastructure. Wang and Shen [2002] stated there were three types of use of the web in supporting design collaboration. In most cases, the web is primarily used by multidisciplinary team members as a medium to share product design data/information/knowledge [Cutkosky, et al., 1996; Roy and Kodkani, 1999; Toye, et al., 1993]. In some cases, the web is used to facilitate product data management and design project management through the integration of the web with some other related technologies [Numata, 1996]. In some other cases, the web may only be used to track and monitor the status of the design process [Shen and Barthes, 1997].

The Madefast developed by Cutkosky, et al. [1996] at Stanford University is one of the early examples of using the web extensively for design collaboration. In the Madefast, a web-based shared and comprehensive online documentation is created to contain a mix of formal design information (including geometric models, diagrams, analyses and test results) and informal information in the form of emails, sketches, photographs, and video clips, and each participating group is responsible for their own part of the documentation. The documentation is decentralised, accessible and searchable to all design participants. Roy and Kodkani [1999] developed a web-based collaborative design environment with various levels of abstracted product models. In their system, a centralised database located in the web server side is responsible for storing and sharing the product models that each designer creates using conventional CAD software. All these product models, upon requested, can be decomposed at the face and feature levels and be represented in VRML[9] formats in the web pages.

[9] VRML: The Virtual Reality Modeling Language (VRML) is a standard file format for representing 3-dimensional (3D) interactive vector graphics, designed particularly with the World Wide Web in mind. It completely supports 3D models with polygonal rendered objects, materials, etc. VRML supports the hyper linking feature. Models developed using VRML can

Su, et al. [2004, 2005] developed a Web-Enabled Environment (WEE) to facilitate online EU-China design collaboration. The WEE supports remote execution of large size executable programs (RELSP), product data file exchange between different CAD systems through a neutral file format, and data sharing with distant users. In the research work presented by Huang, et al. [1999a, 1999b], ActiveX was used to develop a morphological chart-enabled collaborative conceptual design environment and web-based DFX (Design For X) tools.

Web-based collaborative product design systems usually employ a special kind of client/server architecture with a thin client. They often use ODBC/JDBC technologies for integrating design databases to support design information sharing over the web. A detailed discussion on the issues in the development and implementation of web-based applications for product design and manufacture can be found in Ref. [Huang and Mak, 2001].

Table 2.2 summarises some research frameworks and tools developed to support web-based collaborative product design. It should be noted that quite a lot of frameworks and tools have been proposed and/or developed during the past decade and Table 2.2 just lists a few typical examples among them and it is by no means exclusive.

be linked to other documents on the Web which can contain textual, numerical or audio/visual data.

Table 2.2: A summary of research works on web-based collaborative design

R/D Works	Functional Characteristics	Implementation Technologies
Madefast [Cutkosky, et al., 1996]	An online documentation system for posting, sharing of design information and data, accessible and searchable on the desktop, no central authority	Web, HyperMail, and etc.
WEE [Su, et al., 2004, 2005]	Remote execution of large size executable programs, product data file exchange between different CAD systems, and product data sharing	Web, CORBA, neural network, genetic algorithm
CPD [Roy, et al., 1999]	A centralised product design database, shared product models with various abstracted levels, VRML-based visualisation	Web, VRML, PERL, MS Access, ODBC
DFX shell [Huang, et al., 1999a, 1999b]	A web-based framework integrating various DFX tools using ActiveX, client-server architecture	Web, ActiveX, HTML, CGI
WebCADET [Rodgers, et al., 1999]	An online design knowledge-based system for supporting design decision making and product design evaluation	Prolog, Pro-Web server toolkit, CGI
DOME [Pahng, et al., 1998]	A web-based product modelling and evaluation framework, design collaboration supported by exchanging services	Web, Java Applet, CORBA
KaViDo [Tamine, et al., 2003]	A web-based system for recording and documenting design processes, managing the competences of distributed experts, and exchanging user experiences in order to assist product development.	Web, Java Servlet, Java Applet, SOAP, XML, JSP
WEB-DPP [Xiao, et al., 2001]	A web-based architecture for product realisation process, separated information flows, shared design process database	Web, XML, Java, CGI
OneSpace.net [www.cocreate.com]	A set of lightweight web-based tools including shared project workspace, meeting center, product design model explore	Web, XML, SOAP, Web services, SQL, MS IIS, etc.
ConceptStation [www.realitywave.com]	A web-based interactive collaborative environment for 3D viewing and markup of product models	Web, RealityWare VizStream
Autodesk Streamline [www.autodesk.com]	A web-based online service supporting the publishing, viewing and markup of product models, tracking file versions	Web, RealityWare VizStream, Web services

2.6.2 Client-Server Paradigm

Collaborative design systems are traditionally developed mainly based on the traditional client-server paradigm, where the interaction among processes takes place by means of both message-passing and remote procedure call (RPC)[10]. The enabling technologies include Microsoft distributed COM (DCOM), Java's Remote Method Invocation (RMI), Common Object Request

[10] Remote procedure call (RPC): It is a software protocol that allows a computer program running on one computer to cause a subroutine on another computer to be executed without the programmer explicitly coding the details for this interaction. When the software in question is written using object-oriented principles, RPC may be referred to as remote invocation or remote method invocation. Microsoft DCOM was developed based on it. CORBA, Java's Remote Method Invocation (Java RMI) API, Microsoft .NET Remoting service offered a similar paradigm.

Broker Architecture (CORBA), etc. Research projects in this category aim to provide design collaboration functionalities like those provided by agent-based collaborative design systems.

Table 2.3: A summary of some typical collaborative design systems based on the traditional client-server paradigm

R/D Works	Functional Characteristics	Implementation Technologies
[Li, et al., 2002, 2004]	A manipulation client + modelling server infrastructure, feature-based design, design information updated and broadcast through a event-driven and call-back mechanism	Java RMI, Open CASCADE
CADDAC [Ramani, et al., 2003]	A three-tier (client-server-database) architecture, supporting collaborative shape design, a centralised geometry kernel and constraint solver located on the server	Java RMI, JDBC
PCDS [Hu, et al., 2005]	An client-server architecture based on J2EE and engineering database, collaborative product design facilitating mass customisation	J2EE, CORBA, Web, JSP, Java Servelet
[Mervyn, et al., 2004]	A client-server architecture aiming to support integrated product and process design, a central product modelling server,	Java RMI, JNI, XML, Web
cPAD [Shyamsundar, et al., 2001]	A client-server based architecture for collaborative virtual prototyping of product assemblies, a central solid modelling server	C++, Java 3D, JDBC, Web
[Wong, et al., 2005]	A client-server architecture, graphical-based design tools, a virtual workspace, design locking mechanism	Microsoft .NET, Visual C#, Web

In a traditional client-server paradigm, a server is a computational component located at a given site and makes available a set of services. A client component requests a known remote server to execute a service via an interaction process. The server performs the required service and may produce a result that will be delivered back to the client. Compared with the agent-based paradigm which has an asynchronous model of communication, the communication between the server and the client are synchronous. This means that the client process, after sending the request to the server, suspends, waiting for a reply.

By far, various collaborative product design systems have been implemented by adopting a client-server paradigm. Li, et al. [2002] presented a client-server environment based on 3D feature-based modelling and Java technologies to enable design information to be shared efficiently among members within a dispersed design team. In their system, design task and clients are organised

through working sessions generated and maintained by a collaborative server. The information from an individual design client during a design process is updated and broadcast to other clients in the same session through an event-driven and call-back mechanism. Ramani, et al. [2003] developed a three-tier (client-server-database) architecture based collaborative shape design system, Computer Aided Distributed Design and Collaboration (CADDAC), with a centralised geometry kernel and constraint solver. In their system, the server side provides support for solid modelling, constraint solving operations, data management, and synchronisation of clients. The client side performs real-time creation, modification, and deletion of geometry over the network. Command objects are transmitted between the client and the server to keep the consistency of the master model and the client model. Hu, et al. [2005] proposed a client-server framework based on J2EE and an engineering database to facilitate collaborative mass customisation product design.

In Table 2.3, a summary of some typical collaborative product design systems developed by adopting a traditional client-server paradigm is given. Their functional characteristics and implementation technologies are briefly described.

2.6.3 Agent-based Paradigm

An agent-based collaborative design system is concerned with how a group of intelligent agents can cooperate with each other or human designers to collectively manipulate design information and knowledge and solve design problems. In such systems, agents are communicative, collaborative, autonomous (or semi-autonomous), reactive and intelligent.

The research in applying agents to industrial problems has been well examined by Parunak [1998, 1999]. He identified the challenges that are being faced by modern industries but can be addressed by agent technologies. These

challenges are increased product complexity, increased product diversity and variety, and increasing distributed product development process. He argued that 'agents are best suited for applications that are modular, decentralised, changeable, ill-structured, and complex'.

In agent-based collaborative design systems, agents have mostly been used for supporting cooperation among designers, providing a semantic glue between traditional tools, or for allowing better simulations [Wang, et al., 2002]. At the same time, they are often integrated with other advanced information technologies, such as the Web, CSCW, and Knowledge Engineering in developing collaborative product design systems.

So far, there have been many research works attempting to develop agent-based product design systems. The PACT [Cutkosky, et al., 1993] experiments show how pre-existing engineering software systems can be combined to constitute a distributed system of integrated design information and services. The PACT employs a federation architecture and uses facilitators and wrappers to encapsulate each component system. The SHARE [Toye, et al., 1993] project conducted at Stanford University, USA, aims to help teams of engineers achieve a shared understanding of their designs and design processes, using agent-based computational tools and services for communication, collaboration, analysis, and synthesis. Its subprojects including First-Link [Park, et al., 1994], Next-Link [Petrie, et al., 1994], and Process-Link [Goldmann, 1996], which attempt to use agents to help multidisciplinary design engineers in tracking and coordinating their design decisions with each other in the concurrent design of aircraft cable harnesses with the support of a Redux agent. The DIDE [Shen, et al., 1995, 1996] is intended to integrate the existing engineering tools, like CAD/CAM tools, engineering databases, knowledge-based systems, simulation systems, or other special purpose computational tools, into a truly open system for engineering design and intelligent manufacturing. The SiFAs [Brown, et al., 1995] tries to

investigate design problem-solving using multi-agent architectures and explore elementary patterns of interaction, communication and conflict resolution by providing elementary functionality specific to design tasks using single function agents. Liu, et al. [2002] presented a multi-agent design environment in which human designers and software agents interact with each other, exchange design information and keep track of design state information to assist with collaborative design and evolutionary design.

Table 2.4 gives a summary of some typical research works focusing on developing agent-based collaborative product design systems. A detailed discussion on issues and challenges in developing multi-agent design systems can be found in Ref. [Lander, 1997; Shen, et al., 2000].

Table 2.4: A summary of some typical research works for agent-based collaborative design

R/D Works	Functional Characteristics	Implementation Technologies
PACT [Cutkosky, et al., 1993]	Integrating legacy engineering software systems as interacting agents using facilitators and wrappers, a federation system architecture	Agent, Web, KQML, KIF, CORBA
SHARE [Toye, et al., 1993]	Agent-based tools for product design information capturing, structuring and sharing, federation architecture, asynchronous communication using emails	Agent, Web, KQML, NoteMail, ServiceMail
DIDE [Shen, et al., 1996]	Integrating existing engineering tools into a truly open system for engineering design, exchanging design data and knowledge among agents via a local network or Internet	Agent, Web, Lisp, MOSS
SiFAs [Brown, et al., 1995]	Using single function agents to explore elementary patterns of design interaction, communication and conflict resolution	Agent, Web, CLIPS, KQML
[Liu, Tang and Frazer, 2002]	A multi-agent system architecture to facilitate the exchange of design information and tracking of design state, evolutionary design	Agent, KQML, genetic algorithm
CODA [Li, et al., 2002]	A proactive engineering data management service for collaborative design environments, XML-based structured data repository, central service, distributed data management	Agent, Web, XML, Java
CLOVER [Zhao, et al., 2001]	Improvement of the interoperability among heterogeneous engineering applications, supporting higher level dynamic and autonomous cooperation among engineering applications	Agent, Web, XML, CORBA, KQML, Java SDAI, Java RMI
A-Design [Campbell, et al., 1999]	Multi-objective optimization, two tier representation,evaluation-based iterative algorithm, and automated design synthesis	Agent, Web, Lisp
[Fang, Tang and Frazer, 2003]	Integration of legacy CAD software, supporting both synchronous and asynchronous collaboration processes, versioning and locking mechanisms for product data management	Agent, DCOM
Co-Designer [Hague, et al., 1998]	Localised design agents with high degree of authority for design decision-making based on the rich downstream product life cycle information	Agent, Web

2.6.4 Evaluations

In the former sections, three categories of collaborative design systems are identified and reviewed according to the main technologies and their system architectures.

Web-based collaborative design systems can be accessed through a web browser over a network such as the Internet or an intranet. This type of system is very popular due to the ubiquity of the web browser as a client. The ability to update and maintain web-based systems without distributing and installing software on potentially thousands of client computers is a key reason for their popularity. Web-based collaborative design systems are convenient for designers to take on activities including product design share and review, design discussion, customer survey, etc. However, the life-cycle of each collaborative design interaction between a client and a server is only a unidirectional information flow process. This characteristic hinders the efficient and effective broadcast of design changes made by a designer working with a client to designers working with other clients [Li, et al., 2006]. This deficiency makes it very difficult to realise effective interactive design collaboration by a web-based system.

Collaborative design systems based on the traditional client-server paradigm usually aim to facilitate designers in a more interactive design collaboration process. There are some benefits of using the traditional client-server paradigm [Aderounmu, 2004]. First, it provides clean and simple semantics which make the binding of distributed computations easy. Second, it is efficient in that the procedure calls involved are simple enough for the communication arising to be fast. Third, it is capable of providing secure and highly reliable communication. However, in traditional client-server computing, each system has its own client program which serves as its user interface and has to be separately installed on each user's personal computer.

An upgrade to the server part of the system would typically require an upgrade to the client installed on each user workstation. Furthermore, such a type of system is difficult to extend. These deficiencies result in more support cost and decreasing productivity.

Agent-based collaborative design systems attempt to take full advantage of agent technologies and usually provide distributed designers with more collaboration support functionalities, such as design project management, design process coordination, etc. The agent based approach has received great attention in the past years as a promising alternative to the traditional client-server paradigm. The reasons often given for adopting an agent-based approach to develop distributed software systems are linked to their being pro-active object systems and to the simplification of the system architecture. When used appropriately, applications of agent technology can lead to the desired modularity allowing flexible simulation and to better response and improved software reusability. Furthermore, they allow handling ill-structured or rapidly changing situations in a more economical way [Shen, et al., 2000]. Although agent technology has been considered to be very promising for collaborative design systems, most of the systems that have so far been implemented are domain dependent, intended to integrate legacy design tools and still under proof-of-the-concept prototype development stage.

In fact, the above mentioned information technologies used to implement collaborative design systems are complementary. With the increased complexity of product design and the requirements for information exchange among heterogeneous systems, a collaborative product design framework is usually developed by adopting more than one enabling technology while mainly employing one for implementation convenience.

2.7 Summary

This chapter has reviewed the state-of-the-art status of the research in the relevant areas to the development of collaborative product design systems. The main points are as follows:

- An overview of design theories and methodologies is very helpful for better understanding design problems and the design process. Product design is a systematic problem-solving process and it needs the collaborative efforts of multidisciplinary designers or design teams.

- Two main emerging computer enabling technologies, Computer Supported Cooperative Work (CSCW) and intelligent agents, that motivate this research are examined. The detailed review and discussion of various types of agent system architectures help to identify the appropriate agent system architecture to be used in this research.

- The representation of design knowledge in a collaborative design system is very significant in terms of modelling product data and product design process in an efficient and comprehensive way. Various existing computer-based methods for modelling product data and product design process are reviewed and their limitations are also discussed. The detailed review and analysis of those methods help to identify the appropriate knowledge representation schema to be used in this research.

In summary, collaborative design is becoming a common practice for product development. But the technology for supporting such a common practice is still evolving without forming a solid basis. Tracing the history of development of computer aided design technology, it is not difficult to find out that traditional design tools are developed without a designer (user) centred

approach. Product data modelling and product process modelling in particular have been researched based on a generic approach in which the process is only seen from an information point of view. This has resulted in serious limitations because of the information explosion with the introduction of Internet. New collaborative design tools must therefore be developed to limit or control the available information relevant to specialists and designers, based on a study of collaborative design behaviour. In the meantime, new approaches including agent technology have to be developed based on a methodology which does not entirely reject the existing tools and systems used by designers and engineers.

In this research, a new agent-based framework is proposed to support collaborative product development by improving the efficiency and effectiveness of the management and coordination of a collaborative product design process. The remaining chapters present how such a framework is developed and evaluated.

Part III

Research Proposition

Part three consists of five chapters that describe the proposed agent-based collaborative product design framework. Chapter 3 gives an overview of the system structure of the proposed agent-based collaborative product design framework. Chapter 4 describes the proposed Collaborative Product Data Model (CPDM) and Chapter 5 presents the proposed constraint-based Collaborative Design Process Model (CDPM). Chapter 6 gives the technological details of the implementation of a prototype system. In Chapter 7, real collaborative design examples are used to demonstrate the applicability of the implemented prototype system.

Chapter

3

Towards An Agent-based Collaborative Design Framework

3.1 Introduction

An integration of single user based design tools to support a collaborative design team is less efficient than a framework in which software agents are activated to support the coordination of a design team is the basic assumption of this thesis. As such, a computational model of design collaboration is needed for the development of a system architecture in which various software agents can be identified and tested. This chapter describes a theoretical framework for developing agent-based design systems, with the focus on the management and coordination of a collaborative design process. A Collaborative Product Data Model (CPDM) and a constraint-based Collaborative Design Process Model (CDPM) are proposed to form the key for the software agents to interact with each other and human designers, in an open and distributed computational framework.

3.2 Design Management and Coordination

In the process of product design, any change in product requirement, specification caused by various reasons including misunderstanding among team members can cause delays and setbacks for the project. The current available design tools do not address the issue of design management and coordination of the collaborative design project in a consistent way that is tightly integrated with product data models.

In Section 1.1 of Chapter 1, several key characteristics of a large scale design process are identified. With respect to the management and coordination of a product design project, the main issues arising from the collaborative product design process can be summarised as follows:

1) Design management

At the beginning of a product design project, the design information about the product is usually imprecise, incomplete, uncertain, and is likely to be refined or modified at later stages. Therefore, it is hard to predict how the project progresses. The dynamic nature of product exploration and problem specification requires that the product plan and schedule be flexible and have rooms for changes. This is particularly true when the product involves close collaborations with geographically dispersed partners. In fact, a design process is often a process of the project plan and schedule being revised and updated constantly.

2) Design coordination

During the design process, various decisions are made to assign a value to or modify the value of a design variable by each individual participating in the design project. These decisions have consequences and impose additional design constraints to others involved in the design process. In a collaborative product design framework, the impact of individual decision making while

exploring design on others of the design team must be minimized and it should not undermine the overall progress of the project and bring negative influences to the others working on the same project.

In order to address the problems of design management and coordination identified above, a computational model for collaborative product design management and coordination is required and it should be featured by the following characteristics:

- Supporting collaborative product design project planning, schedule execution and coordination of the design process.
- Helping with the re-planning and re-scheduling of the design process as the needs arise.
- Maintaining the consistency of the product data whenever changes take place, and supporting design constraint propagation.
- Helping participants to understand the rationales behind each design decision especially when there is a violation of design constraints.

Based on the discussion of the characteristics of a large scale design process and the above analysis, a computational model of collaborative product design management and coordination is proposed, as shown in Figure 3.1. Four roles, including Design Project Manager, Designer, Product Data Manager, and Constraint Manager, are identified in a computational collaborative design process.

- The **Design Project Manager** is responsible for task description and decomposition, task planning/re-planning and scheduling/re-scheduling, resource management, and tracking of the design process.

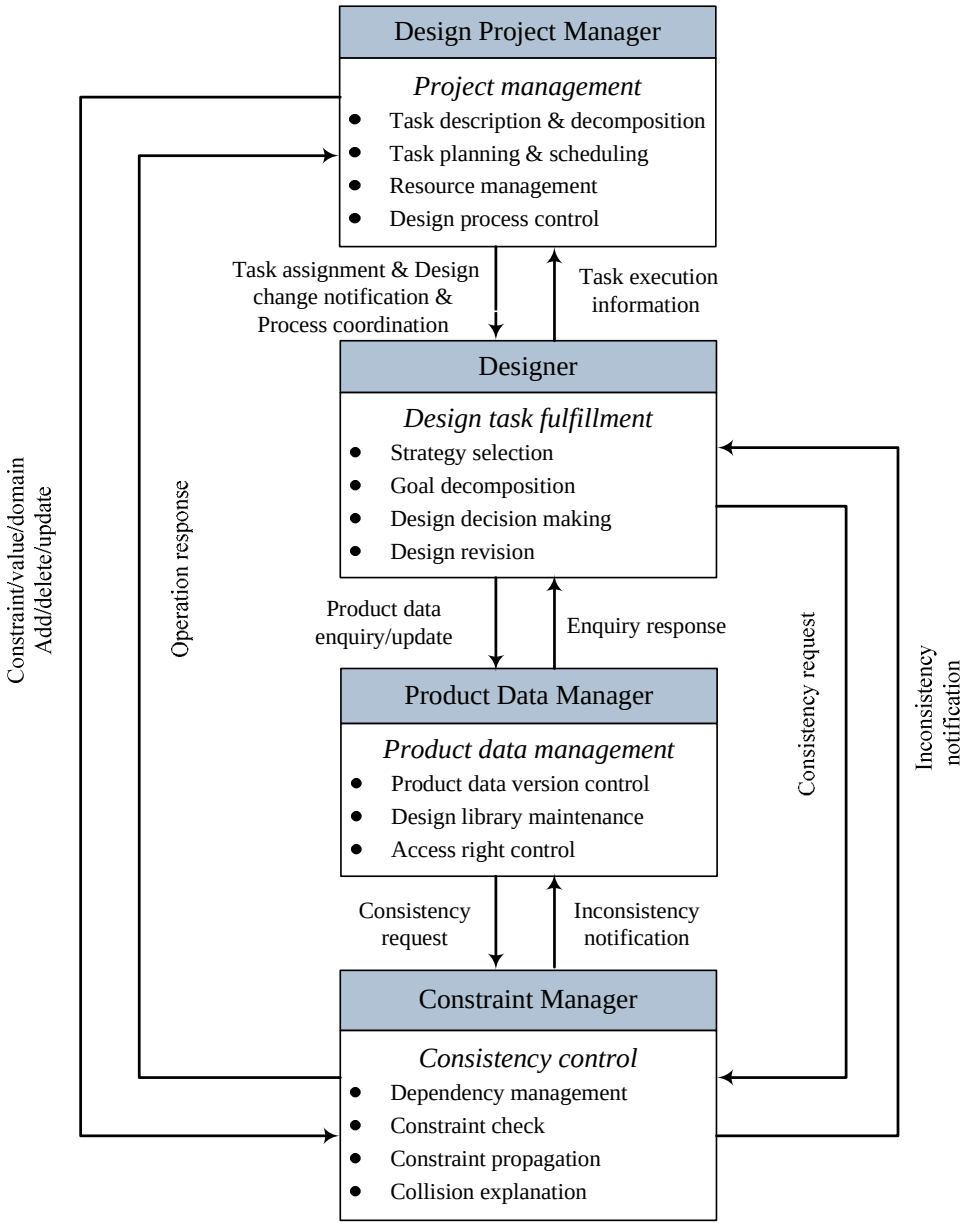


Figure 3.1: The proposed computational model of collaborative product design management and coordination

- The **Designer** serves to adopt the appropriate design strategies to fulfil design tasks assigned by the Design Project Manager and resolve any design constraint violation.

- The **Product Data Manager** is responsible for maintaining design output produced by the designers, maintaining the design library, product data version control, and access right control.

- The **Constraint Manager** is responsible for design dependency management, constraint check, constraint propagation, and collision explanation.

This computational model can be explained in the context of the management and coordination of a collaborative design process in more detail.

To carry out a design project in a collaborative design framework, design tasks must be planned first or at least initiated by someone acting as a design manager, or it will be difficult for the team to start. Suppose a new product design project begins with the preparation work carried out by a design project manager, this may include defining the specification of the design task, decomposing the design task into subtasks, planning and scheduling design tasks, etc. There may not necessarily have any formal procedures to follow even in highly organised firms and enterprises. In a computational process, the design specifications, i.e. the design goals, can normally be defined as a set of design requirements (constraints) that the final design output will be evaluated against. A design task, in a computational form, is represented with a set of attributes including inputs, expected outputs, duration, start time, end time, and assigned agents. Inputs and outputs can be passed in the form of product part/assembly, parameters, which can represent data as well as documents or physical objects. A task can be further decomposed into several subtasks. In this way, the decomposition of the overall design project can be viewed as a process of producing a design task tree although in real design such decomposition may exists in many other forms, such as charts or diagrams.

Design task planning differentiates from scheduling in that planning is to sort out the logical links between inter-related tasks while scheduling focuses on specifying the resources as well as start and end times of the tasks to implement a design plan. The planning and scheduling of all design tasks must make sure that the project progresses in a timely controlled manner, in the

meantime, ensuring the identification of successful design solutions, given the resources and time available. As such, these two activities cannot be formulated as a quantitative search or optimization problem. Rather, they are integrated as a dynamic process in which both quantitative and qualitative constraints must be best resolved by human design managers. These two activities become more important in a collaborative process since the conflicts of interest and commitment become more evident. Therefore, supporting these two activities in a collaborative design support environment is an essential task for the framework.

By determining which of the tasks' input has to be in place before the work on it can be started, and by finding out which tasks produce the required input, the dependency relationship between two tasks can be identified, and then a plan can be created. This plan describes an ordered sequence of tasks and the order is imposed by the tasks' input and output relationships.

In formulating a computational model for collaborative design management and coordination, it is assumed that a causal dependency exists between the available inputs and the produced outputs of a design task. Designers and other resources need to be scheduled by the design project manager to do various tasks of the design plan, by assigning them to tasks with a specified time interval. By assigning a designer and the needed resources to each of the tasks of the plan, the timing of each task is identified and the design schedule is made. The design schedule needs to state the critical path, and for each task its earliest and latest start and end time. Because of the uncertainty at the beginning of a design project, as more information becomes available, the schedule may need to be constantly adjusted.

During the design process, tasks are executed by Designers through making design decisions to assign suitable values to design variables or resolve design constraint violations which finally result in the product design outputs. In the

context of a concurrent engineering design process, shared tasks or overlapping tasks are more difficult to coordinate than isolated tasks. In this research, a constraint-based approach is proposed to facilitate the coordination of the collaborative design process. The Constraint Manager is responsible for managing the consistency of design constraints. For example, when a design decision is made by a Designer to assign a value to a design variable, the Constraint Manager will check the consequences to the consistency of the design constraint network and initiate a design coordination process if required.

The design project manager is responsible for supervising the execution of the scheduled tasks. When a new task occurs, or the input for one of a designer's tasks becomes available or changed, or a task or task assignment is invalidated, or the timing for a task has been changed, notifications will be sent to the concerned designers.

3.3 An Agent-based System Framework

Based on the requirements for a collaborative product design framework identified in Chapter 1 (see Section 1.1) and the previous proposed computational model of collaborative product design management and coordination, an agent-based framework is proposed. It aims to facilitate, rather than automate, the management and coordination of a collaborative product design process where multidisciplinary designers or design teams involved are geographically and temporally dispersed.

The proposed agent-based framework is designed as a network of software agents which interact with each other and the participating team members to facilitate the collaborative design work. These software agents include:

- Project Management Agent,
- Design Agents,

- Product Data Management Agent,
- Design Coordination Agent,
- Design Communication Agents,
- Agent Manager.

The importance of a software architecture is defined by the Carnegie Mellon University's Software Engineering Institute in its glossary [http://www.sei.cmu.edu/opensystems/glossary.html]:

"*The system architecture is in essential a representation of the system in which there is a mapping of the intended functionalities onto the hardware and software components, a mapping of the software architecture onto the hardware architecture, and human interaction with these components.*"

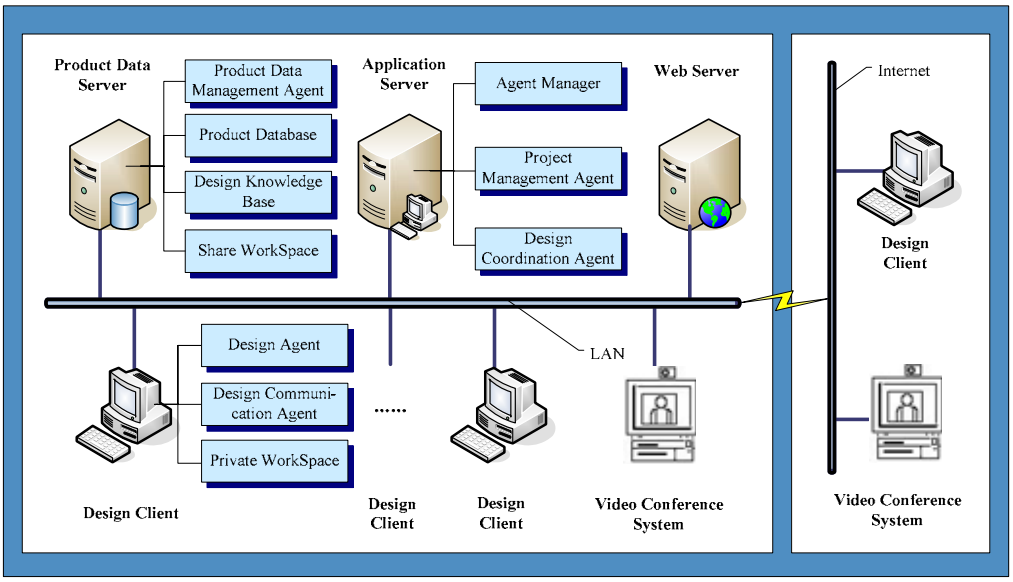


Figure 3.2: The system architecture and hardware configuration of the proposed agent-based framework supporting collaborative product design

The system architecture and hardware configuration of the proposed agent-based framework are illustrated in Figure 3.2. The system architecture is designed to integrate all the software agents with design and engineering

tools/services including product database, design knowledge base, share/private workspace, video conference system, and web, to interact with human designers in an open environment.

The functionalities of the main software agents in the proposed agent-based framework are described as follows:

- The **Project Management Agent** is responsible for helping a project manager to describe the design project, specify the initial parameters and design constraints, decompose a complex design project into sub tasks, assign them to designers, schedule them, and track the whole design process. It has a user interface for assisting the project manager in managing the project plan and schedule, keeping track of progress of the project, and more importantly, making necessary changes while the plan gets more detailed and the higher levels of the plan need to be updated. In addition, it serves to remind the project manager and involved designers of the outstanding issues related to schedule execution.

- The **Design Agent** is a special type of software agent proposed to assist human designers in carrying out product design activities. It integrates some traditional popular CAD systems and is used by the designer to fulfil design tasks through cooperation with other software agents. These agents perform key design tasks and they will be explained in more detail later.

- The **Product Data Management Agent** fulfils the product data manager role in the computational model of collaborative product design management and coordination. It is responsible for managing the product database and ensuring the consistency of the product data in the product design process, and informing concerned design agents

of product data change events (such as submission, modification and so on) made by other design agents.

- The **Design Coordination Agent** serves to maintain the consistency of a network of design constraints. It is responsible for coordinating the collaborative design process through design constraint propagation, notifying involved designers of constraint conflicts, explaining the reasons behind the constraint conflict, and helping designers address the conflicts.

- The **Design Communication Agent** provides several typical means of communication support for the interaction among software agents and human designers. These means of communication include messenger, email, video/audio conferencing, file transfer service, application sharing, whiteboard, coordination message transporting service, etc.

- The **Agent Manager** is responsible for controlling the utilisation and availability of all software agents by maintaining an accurate, complete and timely list of all active agents through which agents residing in the framework are capable of cooperating and communicating with each other.

As shown in Figure 3.2, in the proposed agent-based framework, all the software agents are connected by a local network (LAN) via which they communicate with each other through asynchronous message passing. In addition, the external software design agents, residing on the remote sites carrying out specific design tasks, can communicate with agents located in the local network via the Internet. For the convenience of system deployment and management, the agent manager, the project management agent, and the design coordination agent run on an application server; the product data management agent operates on a product data server which is also responsible for maintaining the product database, design knowledge base, as well as the

shared workspace; the design agent, design communication agent operate on the design client computer on which human designers work.

3.4 Design Agent

Although various agent-based systems have been developed in the domain of product design, design agent as a generic component of an intelligent collaborative design environment has yet to be formally specified, implemented, integrated, and tested.

Saunders and Gero [2001] proposed a curious design agent. It uses a computational process called novelty detection to search the space of two dimensional patterns by a simulated Spirograph for guiding its design actions. The computational model of curiosity employed by the curious design agent seems to work well to search and explore unfamiliar design spaces of a non-routine design task for creative designs, however, it is not applicable to complex product design problems which belong to the domain of routine design and rely much on experiential knowledge. Brazier, et al. [2001] developed a generic architecture for a design agent using the compositional development method for multi-agent systems DESIRE [Brazier, et al., 1995]. In their proposed agent model, strategic reasoning and dynamic management of requirements are explicitly modelled. However, the design model they employed is too abstract and general, and it is difficult to adapt it to real complex product design problems.

In this research, a type of design agent, which is a kind of semi-autonomous and domain-dependent agent, is proposed to support designers in the collaborative design process. It encapsulates traditional CAD software as well as integrating with the design knowledge base. The main components of this design agent include traditional CAD software and its wrapper, collaborative

design tools, the design knowledge base, a private workspace, and the communication layer (shown in Figure 3.3).

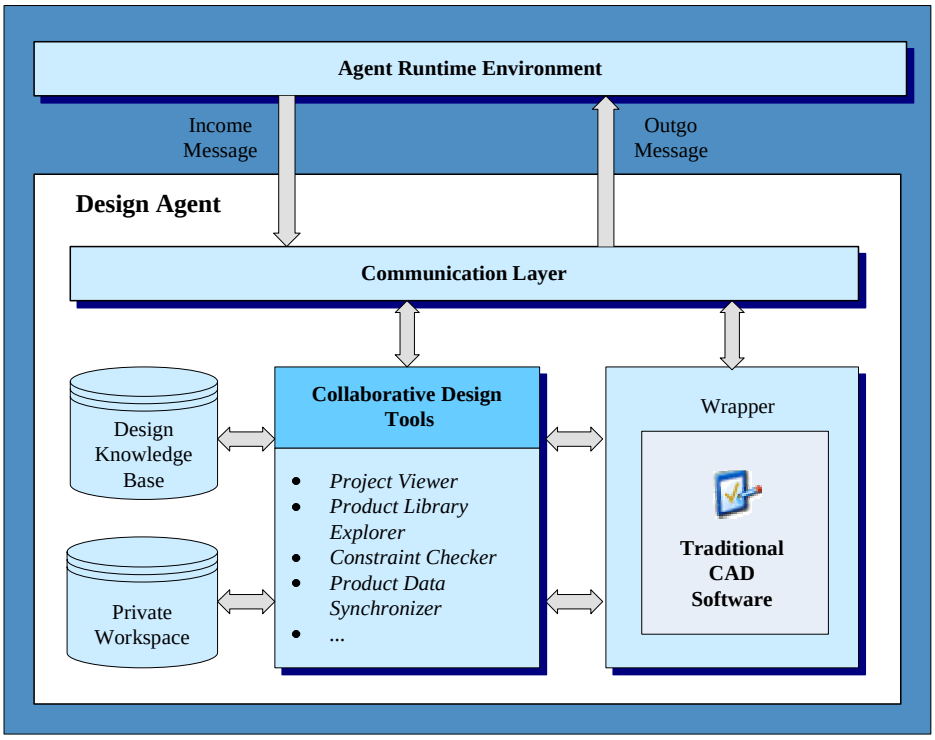


Figure 3.3: The main components of the proposed design agent

- **Traditional CAD Software** is encapsulated into the design agent using a wrapper. There have been some standards, such as CORBA, Microsoft DCOM, allowing legacy applications to be encapsulated using wrapper classes and to behave as distributed components. There are three advantages of including CAD software in a design agent. First, traditional CAD software is mature and can provide powerful solid modelling capabilities. Secondly, the extra burden imposed on designers to get familiar with the collaborative design system can be kept to a minimum.

- **Collaborative Design Tools** assist designers in examining information related to project management including design task planning and scheduling, exploring the product data library, providing multiple

design constraint views, checking the consistency of design constraints, synchronizing product data, etc., with the cooperation of other software agents in the proposed agent-based framework.

- The **Design Knowledge Base** is used for maintaining design knowledge and context knowledge. The design knowledge includes problem solving strategies, design rules, functions, methods, mathematical models whilst the context knowledge includes the information relevant to the background and the design context in terms of antecedents and consequences to which the design knowledge is applied.

- The **Communication Layer** provides a communication mechanism to facilitate the interaction with other software agents through the agent runtime environment. As the FIPA-ACL (Agent Communication Language)[11] has been widely used in various agent-based systems to support communication processes among software agents, and has been accepted as one of the standard agent communication languages, it is adopted as the communication language of the proposed agent-based collaborative product design framework.

When the designer is interacting with the design agent to carry out a design task, if there is any incoming message received from the communication layer, the incoming message is processed and a certain type action will be taken according to the current design context and the design knowledge stored in the design knowledge base. For example, when a design constraint violation occurs, a designer who is responsible for a design variable involved in the concerned design constraint would receive an incoming message requesting a

[11] FIPA-ACL: Agent Communication Language (ACL), proposed by the Foundation for Intelligent Physical Agents (FIPA), is a proposed standard language for agent communications. An agent communication language provides language primitives that implement the agent communication model. Knowledge Query and Manipulation Language (KQML) is another proposed standard.

compromised design. Then, this designer can choose to reject the request, or choose to deal with the design constraint conflict collaboratively with other design agents and designers.

It should be noted that these design agents are not intended to automate the design process, but provide support to human designers in terms of design knowledge, constraint propagation, conflict resolution, and design process coordination through cooperating with other software agents in the proposed agent-based framework.

3.5 Knowledge Representation

Product design is a knowledge intensive process in which sharing and reusing various kinds of knowledge involved in the product life-cycle among multidisciplinary designers is crucial. A systematisation of design knowledge representation is essential for utilising general design problem solving methodologies and domain-specific design knowledge throughout the design process.

During the product design process, there is a large amount of design knowledge that designers may call upon and utilise. The ability of a group of distributed designers to identify the design problem and its solutions is based on well organised and exchangeable design knowledge.

The knowledge necessary for design support is considered to have three categories as follows:

- Static design knowledge representing design objects and concepts (domain knowledge);
- Heuristic and inferential design knowledge representing design problem solving strategies and methods (design knowledge) that can

be invoked by different types of users as design knowledge sources for a range of design tasks; and

- Dynamic knowledge (new knowledge) generated during the design process when applying previous two categories of design knowledge as a result of carrying out a design task.

The static design knowledge consists of design objects and common concepts in the domain of a design application and it is mainly used to build product data models. A product data model describes the product data structure and contains descriptive product information including specification, functions, geometry, behaviour, attributes, variables, constraints, etc. The heuristic and inferential knowledge is mainly for exploring the solutions of design problems. It is knowledge about the design process and design problem solving. This kind of design knowledge is often used to manipulate static design knowledge to generate the knowledge of a new design and it can be described in two levels: design activities and design rationales.

The importance of a better representation for design knowledge in the development of computer-aided design systems has been recognised by many individuals and organisations as a key solution towards future competitive advantages in product development. However, few research endeavours have been found in developing a unified design knowledge representation schema for both static product data knowledge and dynamic design process knowledge as reviewed in Section 2.4 and Section 2.5. In this research, in order to model the constantly evolving design process and the rationales resulting from design collaboration, a Collaborative Product Data Model (CPDM) and a constraint-based Collaborative Design Process Model (CDPM) are proposed to capture and represent the collaborative design knowledge. Next, these two proposed models are described briefly.

3.5.1 CPDM

It has been analysed in Section 2.4.2 that the existing product data models suffer from a number of limitations and they lack required information to support collaborative design in terms of design management and coordination.

In this research, in order to provide a seamless collaborative product design framework, a Collaborative Product Data Model (CPDM) is proposed in the agent-based collaborative product design framework under consideration. The CPDM is established by extending the traditional CAD product data model that is usually a parametric feature-based model employed by most popular CAX (CAD/CAE/CAM) systems. As illustrated in Figure 3.4, at an abstract level, the proposed CPDM is the combination of three data modules: Collaborative Design Management Data, Product Design Coordination Data, and Product Data.

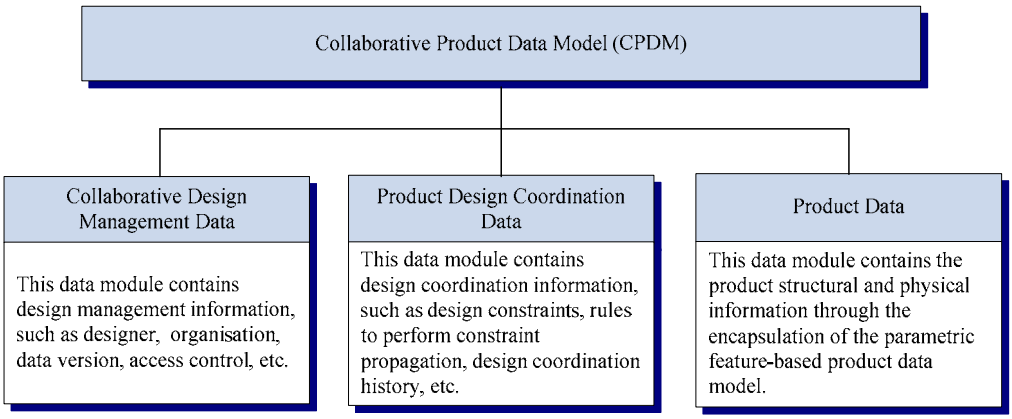


Figure 3.4: The main data modules of the proposed Collaborative Product Data Model

- **Collaborative Design Management Data** contains the information required for managing and maintaining product data in a collaborative product design process. Such information includes information about product designer, information about the organisation, product data

version control, accessing and modifying permission, and information about the alternative strategies or processes.

- **Product Design Coordination Data** contains the information required to coordinate the collaborative product design process. Such information includes user-specified design constraints, rules to perform constraint propagation, and recording design history.

- **Product Data** contains the product structural and physical information through the encapsulation of the parametric feature-based product data model.

3.5.2 CDPM

In a collaborative product design process, the product is simultaneously designed by multidisciplinary designers, each of whom is working on different parts of the product and may consider the design problem at hand from various life-cycle perspectives. The design specifications (requirements) and all these perspectives put forward various design constraints that the final design must satisfy. In this way, the design constraints provide a common language for all the designers to exchange design ideas with each other.

In this research, a constraint-based Collaborative Design Process Model (CDPM) is proposed to work jointly with the Collaborative Product Data Model (CPDM) which is introduced in the previous section to facilitate the management and coordination of a collaborative design process. The CDPM views a collaborative design process as a sequence of state transitions from one design state to another resulting from design decisions made by participating designers and the precondition of the design state transition is the consistency of all design constraints being preserved. In the CDPM, the design constraint propagation and conflict notification are realised through a complex, dynamic, and hierarchical design constraint network.

In the CDPM, a number of main concepts are identified and they are design objective, design product, design variable, design constraint network. Next, these concepts are briefly introduced:

- The *design objective*s describe the specification of the product to be designed. The final product design must meet and be evaluated against these design objectives.

- The *design product* is the product currently under design in order to meet all the design objectives. Each participating designer works on a part (component) of the design product and all involved designers cooperate with each other to fulfil the design tasks.

- A *design variable* is an independent variable or dimension which forms part of the description of a product design and it is directly under the control of a designer. Typical design variables include thickness dimensions, geometrical dimensions, values of material constants, etc.

- The *design constraint network* is a set of design constraints, each of which represents a design requirement to be fulfilled and is denoted by a relation over a number of design variables.

This constraint-based collaborative design process model is proposed to work with the collaborative product data model jointly to facilitate the management and coordination of a collaborative design process. During the collaborative design process, the consistency of the design constraint network which is stored and maintained in the design coordination data module of the CPDM is monitored all the time. As long as a violation of a design constraint arises as the consequence of a design decision made by a designer, those designers involved are identified through the support of the collaborative design management data module of the CPDM, and then a design coordination process is initiated.

3.6 A Collaborative Design Example

To illustrate the ideas behind the proposed agent-based collaborative product design framework, a collaborative product design example is used here to examine how designers are assisted by it in managing and coordinating the collaborative product design process.

It is presumed that there is a new design project whose objective is to design a dining table seating six persons and the accompanying chairs and a design team consisting of one design manager and two designers (Designer1 and Designer2) is responsible for this design project. The collaborative design process in which the design team is supported by the proposed framework can be described as follows:

Step 1: Using the design project management agent, the design manager starts to decompose design task, plan these tasks, assign them to individual designers, and schedule these tasks. Upon the completion of the task scheduling, task assignment messages are sent automatically to corresponding designers from the design project management agent which then begins to monitor the execution process of these tasks. Suppose that the task of designing the dining table is assigned to the Designer1 and the task of designing the chair is assigned to the Designer2.

Step two: Upon receiving the design task, the designer attempts to find a suitable product design from the product data repository using the product data management agent. In this example, he/she would select a dining table or a chair. If there is no suitable match for the current design task, the designer creates a new product model using the design agent.

Step three: When a designer is working on a product model, he/she can add a design constraint by specifying a relationship among the design variables of the same part or different parts of the product using design agent from time to

time. As for the dining table and chair design example, for ergonomic reasons, a design constraint DC1 can be specified between design variables: DV1 (the height of the dining table top surface) and DV2 (the height of the chair seat) stating the vertical distance between the top surface of the dining table and the seat of the chair should be within a certain range [200, 400] (mm).

Step four: When a design decision is made by a designer, the design coordination event will be triggered and the design coordination agent would automatically check the consistency of design constraints in the design constraint network. If a design constraint is found to be violated, the design coordination agent would notify involved designers of the constraint violation and its reasons. As for the dining table and chair design example, if DC1 is found to be violated, Designer1 and Designer2 would be notified.

Step five: After the involved designers negotiate with each other through the design communication agent, a new design decision is made by a designer. At this time, if no constraint violation is found by the design coordination agent, the product design process state is updated to reflect the design decision by the product data management agent.

Step six: The collaborative design process continues until all design subtasks are finished and a desired product design is achieved.

It should be mentioned that the design example used here is only briefly introduced and tries to give a general overview of how designers can be assisted by the proposed framework in managing and coordinating the collaborative product design process. This design example will be further elaborated, in conjunction with the detailed explanations and descriptions of the proposed CPDM and CDPM as well as the implemented prototype system in the following chapters.

3.7 Summary

This chapter has given an overview of the proposed agent-based collaborative product design framework. The main points are as follows:

- Based on the discussion of the characteristics of a large scale product design process, a computational model of product design management and coordination is proposed.

- A multi-agent system architecture of collaborative product design is proposed. The main components of the framework and their functionalities are described in detail. The system architecture is designed as a network of software agents which interact with each other and participating team members to facilitate the collaborative design work.

- A special type of design agent is presented to encapsulate traditional CAD software, as well as the design knowledge base aiming to assist designers in the collaborative design process.

- A computational representation of design knowledge is introduced. A Collaborative Product Data Model (CPDM) and a constraint-based Collaborative Design Process Model (CDPM) are proposed and briefly described. They are proposed to capture and represent the constantly changing design process and the rationales resulting from design collaboration.

In the following two chapters, the proposed Collaborative Product Data Model (CPDM) and the constraint-based Collaborative Design Process Model (CDPM) will be fully explained.

Chapter

4

Collaborative Product Data Modelling

4.1 Introduction

Product data modelling is one of the key issues to be tackled in the development of computer-aided design systems. In the context of developing computer-based design systems, product modelling deals with the internal representation of a product in terms of its function and form as it is used in many of the traditional CAD systems. However, in order to support the process of design collaboration, additional information needs to be embedded within the product data to facilitate a communication process during which the change of attributes of a product may affect others involved in the collaboration. This introduces added complexity in the representation of product data, which must contain additional variables and constraints for concurrent operations which may occur asynchronously. In this research, a Collaborative Product Data Model (CPDM) is proposed to deal with this problem. The CPDM is intended to work with the Collaborative Design Process Model (CDPM) which is proposed to model the constantly changing

collaborative design process. This chapter discusses the CPDM in detail before introducing the CDPM in the next chapter.

4.2 The Requirements

Product data modelling is concerned with the definition, representation, manipulation, and management of complex product data elements as well as various relationships among them. During a collaborative design process, the design constantly evolves and the attributes of the product data model as well as the relationships among them are always changing. Therefore, a computer-aided product design environment should not only support the representation of the end product and the relationships between its components, but also the evolution of the components and associated knowledge over the whole life cycle of the product [Shen and Barthes, 1995].

In the preface for a special issue of the journal *Computers in Industry* that was dedicated to papers on reporting the research of product data and product design process modelling, Pels [1996] gave two reasons for the development of techniques for product data modelling. The first is to aid the designer in imagining and validating the new product development. The second is to communicate the product development process, if that is another person than the designer. It is even more the case when the product to be designed becomes more complicated and multidisciplinary designers are involved in the product creation and development process with the assistance of a collaborative product design framework.

In Chapter 1 (see Section 1.1), a number of characteristics of a large scale distributed design process are identified. Compared to the problem of product data modelling in a traditional stand-alone computer aided design system, there are new requirements for product data modelling in a collaborative design framework. These new requirements can be summarised as follows:

- Design project management: The design of real world complex products usually needs the cooperation of multidisciplinary designers using various heterogeneous engineering design tools. These designers perform design activities either synchronously or asynchronously. Design project management data must be represented in order to ensure the management and coordination of the collaborative design process such that the right product design information is delivered to the right person at the right time.

- Design rationale: In a real world collaborative design project, various design tasks are carried out by different designers to design different parts of the product. The design rationales behind key design decisions are required to be captured and represented in order to promote the product data sharing and the cooperation among designers.

- Constraint propagation: Design constraints are the computational representation of design requirements and they play an essential role in the design process. These design constraints must be represented in a flexible form and their consistency must be kept all the time.

- Version control: In order to represent the evolution of a product design, not only the current state of the product design, but also some of its past states need to be recorded. There are also some circumstances when a few versions of the product design are viewed as different design alternatives.

To provide a data structure which can meet these requirements, a new product data model is needed, when considering it operating within an agent-based collaborative product design framework.

4.3 CPDM – A Collaborative Product Data Model

4.3.1 Overview

In order to overcome the shortcomings of the existing product data models which are reviewed in Chapter 2 (see Section 2.4.2), a Collaborative Product Data Model (CPDM) is proposed to be employed in the proposed agent-based collaborative product design framework. The proposed CPDM is established by extending the popular parametric feature-based product data model and including information required for design management and coordination. As illustrated in Figure 4.1, the CPDM consists mainly of three data modules:

- Collaborative Design Management Data,
- Product Design Coordination Data, and
- Product Data.

Collaborative Design Management Data module contains the information required for managing and maintaining product data in a collaborative product design process. Such a module includes information about the designers, organisational information, product data version control information, accessing and modifying permission, and information about alternative strategies or processes.

Product Design Coordination Data module contains the information required to coordinate the collaborative product design process. Such information includes user-specified design constraints, rules to perform constraint propagation, and information about the design history. Here design history means a record of activities performed during the design process and in particular all the changes that have been made with justifications.

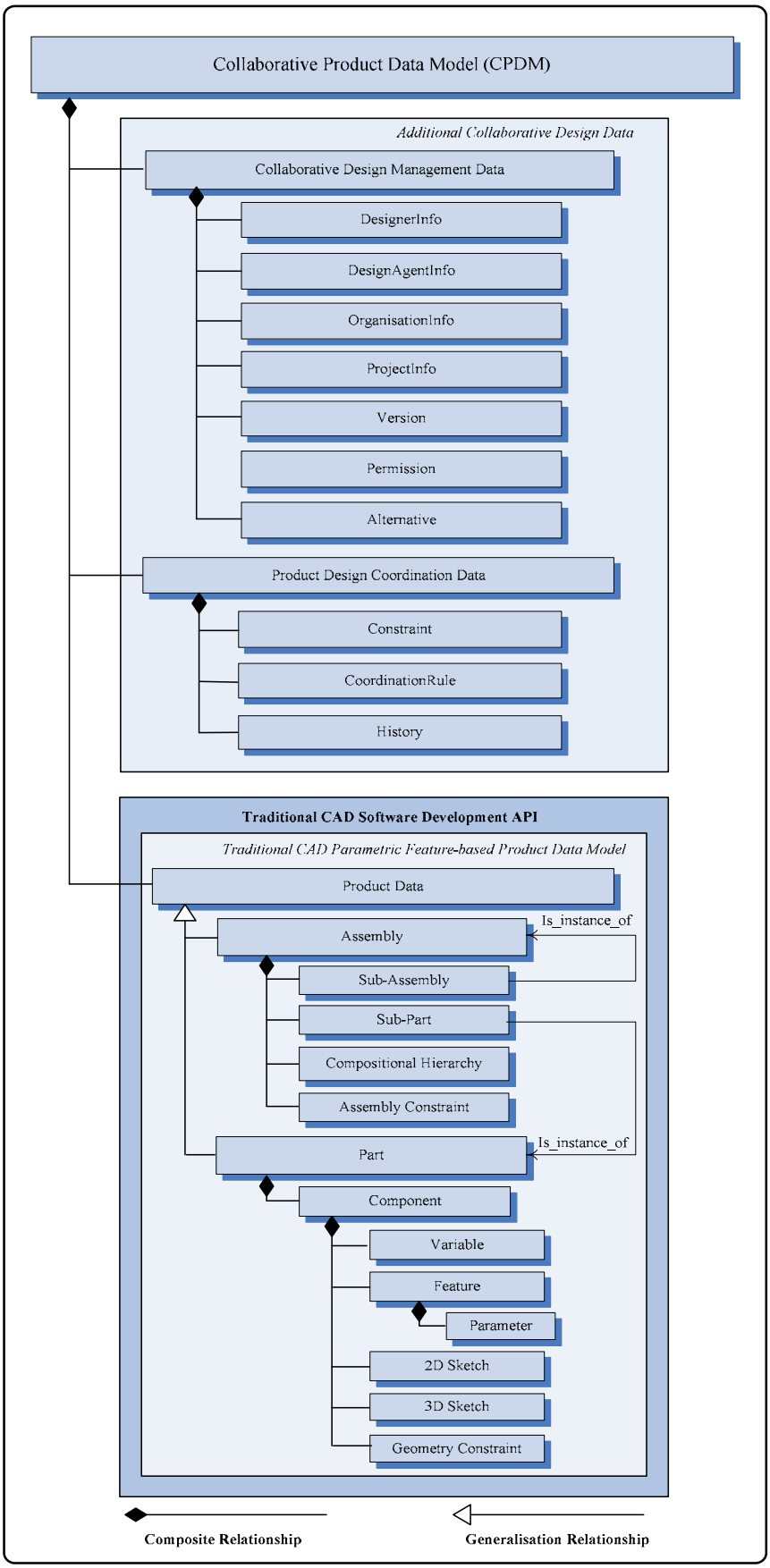


Figure 4.1: The proposed Collaborative Product Data Model (CPDM)

Product Data module contains the structural and physical information of the product through the encapsulation of the parametric feature-based product data model.

In the following sections, the functionalities and the data elements of the main data modules in the proposed CPDM are described in detail.

4.3.2 Collaborative Design Management Data Module

The Collaborative Design Management Data module contains the information that is helpful for managing and maintaining the product data in a collaborative product design process. The detailed data structure of this data module including data elements and their data types is described in Table 4.1. Seven types of information are identified in this data module and they are:

- DesignerInfo,
- DesignAgentInfo,
- OrganisationInfo,
- ProjectInfo,
- VersionInfo,
- PermissionInfo, and
- AlternativeInfo.

DesignerInfo consists of the information about the creator and the design manager of the product part, such as name, email address, and phone number. There are many circumstances in which this information should be identified so that the design coordination activities in the collaborative design process can be performed. For example, when there is a design constraint violation

occurring and this design constraint involves a number of design variables that are under the control of more than one designer, then the information about these designers is required so that the design constraint violation message can be delivered to them and a design coordination process can be initiated among the involved designers.

Table 4.1: The detailed data structure of the Collaborative Design Management Data module

Information Field Name	Data Name	Data Type	Description
DesignerInfo	DesignerName	*String*	The name of the designer in charge of the product data.
	DesignerEmail	*String*	The email address of the designer.
	DesignerPhoneNo	*String*	The phone number of the designer.
	ManagerName	*String*	The name of the design manager in charge of the design task.
	ManagerEmail	*String*	The email address of the design manager.
	ManagerPhoneNo	*String*	The phone number of the design manager.
DesignAgentInfo	Software	*String*	The CAD software used to carry out the design work.
	OperatingSystem	*String*	The operating system on which the design is carried out.
OrganisationInfo	Department	*String*	The name of the department that the designer belongs to.
	Company	*String*	The name of the company that the designer belongs to.
VersionInfo	CurrentVersion	*String*	The current version of the product data.
	PreviousVersion	*String*	The previous version of the product data.
	NextVersion	*String*	The next version of the product data.
	Status	*String*	The design status of the product data.
	DesignState	*Integer*	The release status for the product data.
	Checker	*String*	The name of the person who checked the product data.
	CheckDate	*Date*	The date that the product data was checked.
	CreatedWith	*String*	The CAD software used to carry out the design work.
ProjectInfo	Name	*String*	The name of the design project.
	WebLink	*String*	A web site address.
PermissionInfo	GeneralSecurityMode	*Integer*	A security mode for accessing or modifying activities on the product data.
AlternativeInfo	Title	*String*	The title for the product.
	Category	*String*	A category for the product.
	Keywords	*String*	Keywords that can be used to search for related product.
	Comments	*String*	Comments to the product data.

OrganisationInfo includes the information about the relations among a design team. It helps to identify the information about the members of the design team, such as their roles, domains, and contact methods, etc.

VersionInfo is used to record the evolutionary process of the product design. In some circumstances, it serves to differentiate main modifications in the product design and to take advantage of access to its past versions when undesirable changes have been made. In some other circumstances, a few versions of the product design are viewed as different alternatives of the product design. These alternatives can be examined and evaluated later against each other according to the design requirements. This helps designers to find an optimal design output.

ProjectInfo contains the information about the design project, such as the project name, website where distributed designers can view the latest on-line project schedule or exchange ideas about design on a bulletin board.

PermissionInfo is used to restrict accessing or modifying data on the product. In a collaborative product design system, product data is usually shared among multiple designers. It is useful to permit only certain designers to read or modify certain product data, and assign others read-only access to the data, if a well organised design team is to be established.

AlternativeInfo contains the information helpful to identify alternative product parts, such as the category, keywords, etc.

In this data module, an attribute called DesignState data in the VersionInfo field and an attribute called GeneralSecurityMode data in the PermissionInfo field take an enumeration type of data respectively:

DesignState: It has three modes indicating the release status for the product data:

- 0 --- Work in progress. The product design is still under development;
- 1 --- Pending. The product design is complete and waiting for approval;
- 2 --- Released. The product design has already been approved.

GeneralSecurityMode: It has two modes controlling the security level of the product data:

- 0 --- Public full access. Every one can access the product data with read permission as well as write permission;
- 1 --- Password protected. People can only have the read right or full access right of the product data if he/she can provide the correct read or full-access password.

4.3.3 Product Design Coordination Data Module

The Product Design Coordination Data module contains the information required for coordinating the collaborative product design process. The detailed data structure of this data module including data elements and their data types are described in Table 4.2. Three types of information including Constraint, CoordinationRule, and History are identified as follows:

Constraint is the user-specified design constraint relationship among product elements (i.e. features, parts, or sub-assemblies) in different forms, such as range, equation, rule, etc, corresponding to the design requirements. A design variable involved in the constraint is in fact a pointer that refers to the corresponding design variable in the product data module. In this way, the data redundancy is avoided and the data integrity is improved. There are three levels of constraints: global, subassembly, and part. Constraints on the global level are imposed on all the lower subassemblies and parts of the product. Constraints on the subassembly level are imposed on the lower subassemblies

and parts of the subassembly. Constraints on the part level are only imposed on components that compose the part.

Table 4.2: The detailed data structure of the Product Design Coordination Data module

Information Field Name	Data Name	Data Type	Description
Constraint	Constraints	*ConstraintArray*	A custom array for storing design constraints.
CoordinationRule	GeneralMode	*Integer*	The mode indicating the strategy for controlling the design coordination.
	CoordinationLevel	*Integer*	A number indicating how many nearest levels of design managers are informed of the design constraint violation.
History	HistoryRecordMode	*Boolean*	The mode specifying whether the history of design coordination activities may be recorded.
	CoordinationHistory	*CoordinationArray*	A custom array for recording the design coordination activities.

CoordinationRule is the user-specified criteria of how constraint propagation is performed for the constraints, i.e. what person in what way should be informed as well as what action should be taken when a certain design constraint is violated.

History records the sequence of the main design coordination actions carried out in response to constraint violations. It also contains the information related to the involved designers, the time, and the reason.

As shown in Table 4.2, design constraints are stored using a custom array type of data, ConstraintArray, in which each constraint item (ConstraintItem) has six data attributes: index, name, type, expression, status, and description. In a similar way, the history of design coordination activities is also stored using a custom array type of data, CoordinationArray, in which each item (CoordinationItem) representing a design coordination activity has five data attributes: index, constraintname, designer, time, and description. The constraint item and the coordination item are defined as shown in Table 4.3.

Table 4.3: The detailed data structures of the ConstraintItem and the CoordinationItem

Information Field Name	Data Name	Data Type	Description
ConstraintItem	Index	*Integer*	Index for the constraint item
	Name	*String*	Name for the constraint item
	Type	*Integer*	Type of the constraint which is an enumeration type: "1" standing for equation, "2" standing for inequation, "3" standing for rule
	Expression	*String*	Expression for the constraint item
	Status	*Integer*	Status of the constraint enforcement which is an enumeration type: "0" standing for suspended, "1" for active
	Description	*String*	Description for the constraint item
CoordinationItem	Index	*Integer*	Index for the coordination item
	CostraintName	*String*	Name of the constraint whose violation is addressed by the coordination activity
	Desinger	*String*	Names of designers who are involved in the coordination activity
	Time	*Time*	The time when this coordination activity happens
	Description	*String*	Description for the coordination activity

In this data module, an attribute called GeneralMode data in the CoordinationRule field takes an enumeration type of data and it has three modes controlling the design coordination strategy:

- 0 --- No design coordination is required,
- 1 --- Only the designers involved are informed of the design constraint violation,
- 2 --- Besides the designers, their respective design managers are also informed of the design constraint violation. In this case, the CoordinationLevel indicates how many nearest levels of design managers are also informed. For example, as shown in Figure 4.2, design variables including V1, V2, and V8, are involved in the design constraint --- Constraint1, and design variables including V5, V9, and

V10, are involved in the design constraint --- Constraint2. If the CoordinationLevel is set to 1, then besides Designer1 and Designer4, Manager3 and Manager4 are also informed of the violation of the Constraint1. If it is set to 2, then besides Designer1, Designer3, Manager3, and Manager4, Manager1 and Manager2 are also informed.

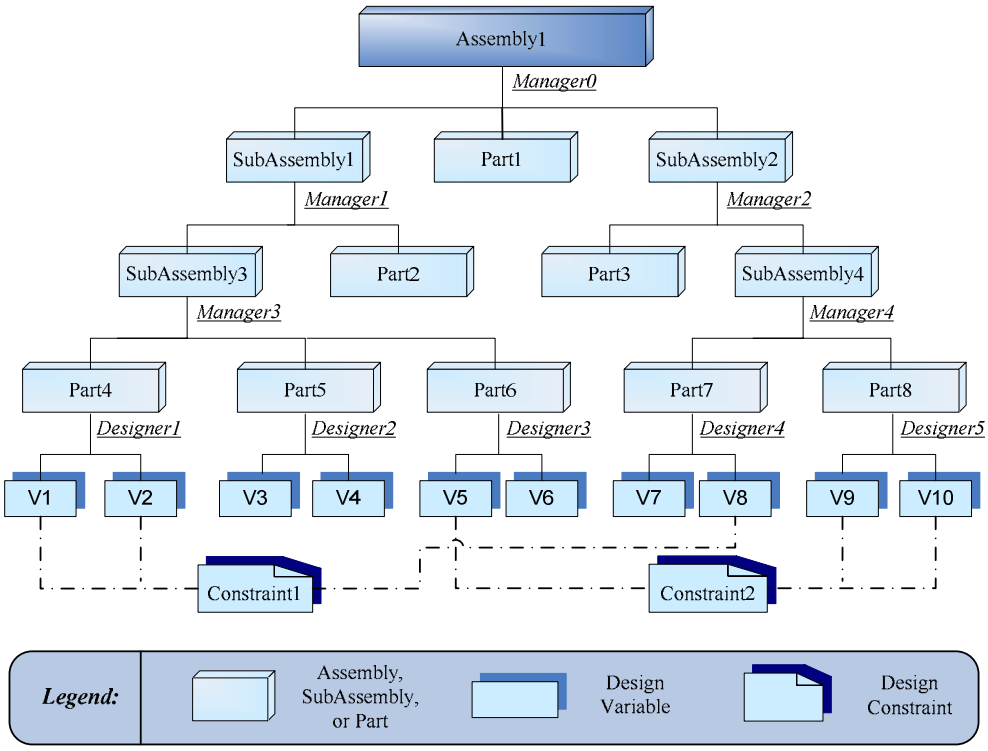


Figure 4.2: The product assembly tree, design variables, design constraints, and the relationship among them

4.3.4 Product Data Module

The Product Data module contains the product structural and physical information. As mentioned previously, CPDM encapsulates the parametric feature-based Product Data Model that is employed by popular traditional CAX software through their Application Programming Interface (API).

As shown in Figure 4.1, in such a product data model, a *Product* is an *Assembly* or a *Part*. An *Assembly* consists of two or more *Sub-Parts* or *Sub-*

*Assemblie*s with *AssemblyConstraint*s applied on them and is structured in a *Compositional Hierarchy*. *A Sub-Part* is itself an instance of another Part and a *Sub-Assembly* is itself an instance of another Assembly. A *Part* has *Component*s, each of which has *Variable*s, *Feature*s, *2D Sketche*s, *3D Sketche*s, and *Geometry Constraint*s. A *Feature* has some *Parameter*s that define its size and shape. A *Variable* with a meaningful name can be defined in terms of an arithmetical combination of parameters.

The selection of the parametric feature-based model as the product data module is based on the following considerations:

- First, after decades of development, the parametric feature-based model has become more and more mature as a standard format for product data sharing and exchange.

- Second, most of the popular commercial CAX systems, e.g. AutoCAD®, Autodesk Inventor®, SolidWorks®, Pro/Engineer®, CATIA®, etc, are parametric feature-based three-dimensional solid modelling systems. By taking the traditional parametric feature-based product data model as the product data model in the proposed CPDM, it is convenient to integrate popular commercial CAX systems into the proposed agent-based framework.

- Third, through the integration of these popular commercial CAX systems into the proposed agent-based framework, time and money are saved and the extra burden imposed on designers to get familiar with the collaborative design system can be kept as low as possible.

4.4 An Example of CPDM

In this section, the use of the proposed CPDM to represent product data in a collaborative design process is described using a design example which is mentioned in Chapter 3.

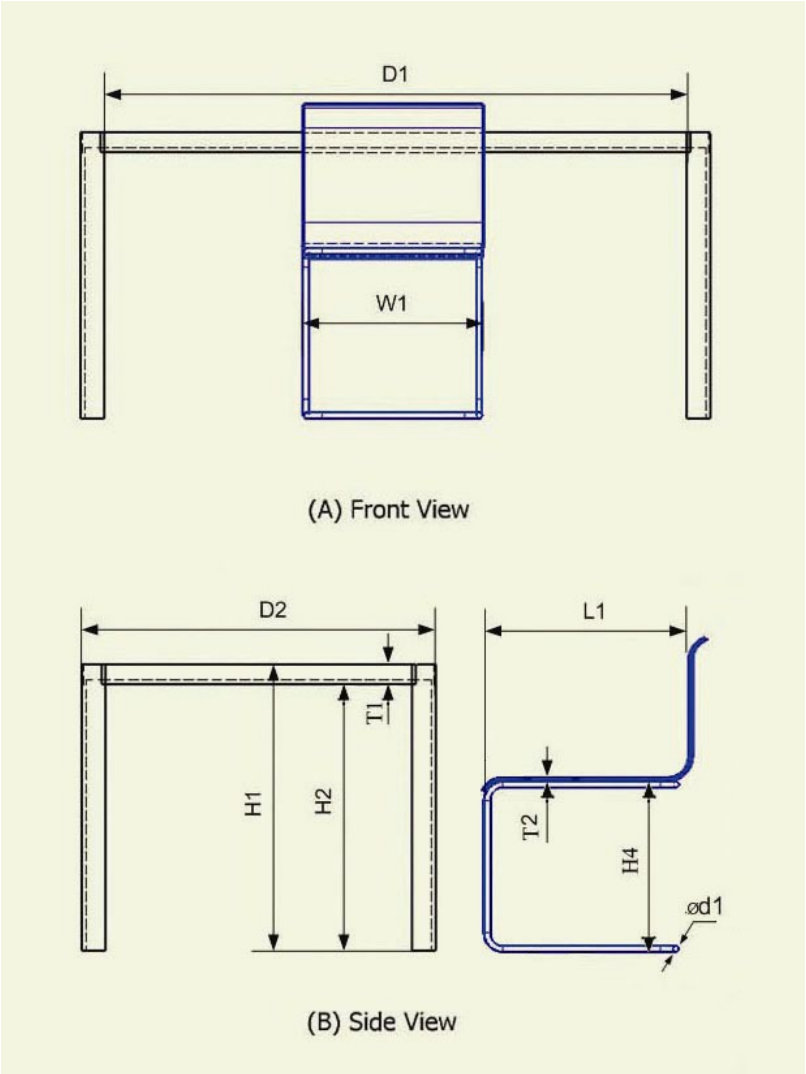


Figure 4.3: The design of a dining table with a chair

As introduced in Chapter 3, a collaborative design example whose objective is to design a dining table with chairs seating six persons is used to validate the applicability of the proposed framework and the CPDM. During the collaborative design process, the detailed product data of a dining table

instance and a chair instance represented using the proposed CPDM, at a certain time, can be described as shown in Figure 4.3, Table 4.4, and Table 4.5.

In this example, a design manager, KathTang, is in charge of the management of the whole design project. One designer, JasonWong, is responsible for the design of the dining table and another designer, LouisLee, is responsible for the design of the chair. At a certain time, the detailed product data of the collaborative design management data module of the dining table instance, and a chair instance is shown in Table 4.4 and the detailed product data of the design coordination data module is shown in Table 4.5.

Table 4.4: The detailed product data of the collaborative design management data module of a dining table instance, and a chair instance

Information Field Name	Data Name	Data Type	A Dining Table Instance	A Chair Instance
DesignerInfo	DesignerName	*String*	JasonWong	LouisLee
	DesignerEmail	*String*	JasonWong@polyu.edu.hk	LouisLee@polyu.edu.hk
	DesignerPhoneNo	*String*	27664444	27664435
	ManagerName	*String*	KathTang	KathTang
	ManagerEmail	*String*	KathTang@polyu.edu.hk	KathTang@polyu.edu.hk
	ManagerPhoneNo	*String*	27764831	27764831
DesignAgent Info	Software	*String*	AutoDesk Inventor 10.0	AutoDesk Inventor 10.0
	OperatingSystem	*String*	Microsoft WinXP	Microsoft 2000
Organisation Info	Department	*String*	Product Design	Product Design
	Company	*String*	DTRC Furniture Factory	DTRC Furniture Factory
VersionInfo	CurrentVersion	*String*	1	2
	PreviousVersion	*String*	0	1
	NextVersion	*String*	2	3
	Status	*String*	Structure design finished	Structure design finished
	DesignState	*Integer*	1	1
	Checker	*String*	KathTang	KathTang
	CheckDate	*Date*	2006-07-14	2006-07-14
	CreatedWith	*String*	AutoDesk Inventor 10.0	AutoDesk Inventor 10.0
ProjectInfo	Name	*String*	Dining Table and Chairs	Dining Table and Chairs
	WebLink	*String*	Http://dtrc.sd.polyu.edu.hk/dining	Http://dtrc.sd.polyu.edu.hk/dining
PermissionInfo	GeneralSecurityMode	*Integer*	0	0
AlternativeInfo	Title	*String*	Dining Table	Dining Chair
	Category	*String*	Table	Chair
	Keywords	*String*	Table, Dining Table, Furniture	Chair, Dining Chair, Furniture
	Comment	*String*	It is designed to sit six adults.	It is designed for adult's use.

Table 4.5: The detailed product data of the design coordination data module of a dining table instance, and a chair instance

Information Field Name	Data Name	Data Type	A Dining Table Instance	A Chair Instance
Constraint	Constraints	*ConstraintArray*	ConstraintArray1 (see Table 4.6)	ConstraintArray2
Coordination Rule	GeneralMode	*Integer*	0	0
	CoordinationLevel	*Integer*	1	1
History	HistoryRecordMode	*Boolean*	TRUE	TRUE
	CoordinationHistory	*CoordinationArray*	CoordinationArray1 (see Table 4.7)	CoordinationArray2

Table 4.6: ConstraintArray1 (Constraints in the design coordination data module for the dining table instance)

Index	Name	Type	Expression	Status	Description
1	Constraint1	Inequation	DT.T1 + DT.H2 >= DC.T2 + DC.H4 + 20	Active	The difference of the height of the dining table and the height of the chair should be more than 20cm
2	Constraint2	Inequation	DT.T1 + DT.H2 <= DC.T2 + DC.H4 + 40	Suspend	The difference of the height of the dining table and the height of the chair should be less than 40cm.
3	Constraint3	Inequation	DT.D1 * 2 >= (DC.W1 + 10) * 6	Suspend	To ensure that all chairs can be packed under the dining table.
4	Constraint4	Inequation	DT.D2 >= DC.L1 * 2	Active	To ensure that all chairs can be packed under the dining table.

Table 4.7: CoordinationArray1 (CoordinationHistory in the design coordination data module for the dining table instance)

Index	ConstraintName	Designer	Time	Description
1	Constraint1	JasonWong	2006-07-07 12:03	The value of the design variable H2 of the 'Diningtable' has been changed from '60' to '70' in response to a constraint violation of 'Constraint1'.
2	Constraint4	LouisLee	2006-07-07 14:33	The value of the design variable L1 of the 'Chair' has been changed from '55' to '50' in response to a constraint violation of 'Constraint4'.

Suppose there are two design specifications that the final design output must meet as follows (the unit of design dimension is centimetre):

1) Due to ergonomic reasons, and to let the users feel comfortably to dine when seating on the chair, the vertical distance between H1 - the height of the top surface of the dining table, and H2 - the height of the seat of the chair, should be within a range [20, 40].

2) The chairs can be packed under the dining table to save room when there is no person using them.

For the design of the dining table, the above stated design specifications can be formulated and defined as four design constraints as shown in Table 4.6 and DT stands for the dining table and DC stands for the chair. For example, in the expression of Constraint1, T1 and H2 are design variables of the dining table and they can be defined using the design parameters which are stored in the product data module.

4.5 Summary

This chapter has given a detailed description of the proposed Collaborative Product Data Model (CPDM) which is employed in the proposed agent-based framework supporting collaborative product design. The main points are as follows:

- A number of requirements for product data modelling in a collaborative product design framework are identified and discussed.
- The hierarchical data structure of the proposed CPDM is described in detail and it is established by extending the popular parametric feature-based product data model and including information required for design management and coordination.
- The functionalities and the data elements of the main data modules including the collaborative design management module, the design coordination data module, and the product data module are elaborated respectively.

In the next chapter, a constraint-based Collaborative Design Process Model (CDPM) will be proposed and presented in detail.

Chapter

5

Collaborative Product Design Process Modelling

5.1 Introduction

A collaborative design process involves constant interaction and communication during which each party of the collaboration team has their own expectation of the progress, and sometimes they have to coordinate with each other in order to proceed. Any frustration in this process will bring negative effects to the outcome of the design. Therefore a computational process of design collaboration must facilitate clear understanding of the expectations of others by communicating the ideas of each with maximum clarity with assigned duties and expected outcomes. With the participation of software agents including design agents, there is a need for a process model supporting the smooth progression of design projects with functionalities to resolve conflict of interests or misunderstanding of intention. This chapter presents a collaborative design process model which operates on the collaborative product data model as presented in the previous chapter.

5.2 The Motivations

A large complex product design project carried out in a collaborative design system often involves various multidisciplinary people and teams who must coordinate their activities based on information flow, available resources, and other constraints. Moreover, because design is a non-deterministic process, it is often necessary to capture incomplete information that evolves over time. A formal and comprehensive method for modelling design process is of much interest to the development of computer supported collaborative design systems. According to Park, et al. [1999], there are at least three important motivations for seeking such a method and they can be summarised as follows:

- *Design process automation:* While collaborative design processes tend to benefit from computer support systems, they are too complex and extend over too wide a range of functions to apply local automation without incurring side effects. A formal formulation of collaborative design processes is required for (a) addressing various problems arising from missed or misunderstood dependencies between process components, (b) selecting and designing local design process automation, (c) understanding and coordinating the collaborative design work, and (d) defining a conflict management strategy.

- *Reduction of design process cycle-time:* The global market competition has driven product manufacturers to reduce the time-to-market cycle time through Business Process Reengineering (BPR) and other means. Increasingly, design processes that previously were conducted serially are now performed simultaneously, thus put much greater demand on effective and efficient communication and coordination.

- *Coordination of distributed activities:* Advancements in computer network technology offer new ways for design engineers and other

specialists to work over the network. Product design activities are becoming increasingly carried out using computer tools in a distributed fashion in all stages of product development. This has increased both the need for and difficulty of generating, maintaining and sharing a common understanding of events among collaborators [Cutkosky, et al., 1996]. Furthermore, as design information tends to become dispersed in multiple heterogeneous systems, the problem of coordination for individual design tasks is becoming expedited.

5.3 CDPM – A Collaborative Design Process Model

5.3.1 Overview

As reviewed in Section 2.5, there are a number of research projects on the modelling of product design process. Most of them are activity-based models which try to represent the relationships between tasks and focus on the dependencies pertaining to the information flow. However, they suffer from a number of limitations that make them insufficient for the development of computer supported collaborative design systems. An obvious disadvantage of this type of model is that they do not provide satisfactory solutions to resolve design constraints (or conflicts) which occur so often in design collaboration.

In this research, to overcome the shortcomings of the existing methods for design process modelling, a constraint-based Collaborative Design Process Model (CDPM) is proposed to model the constantly changing design process and the rationales that result from design collaboration.

Design constraints are a common computational representation of product design requirements. By using design constraints to express the relationships among design variables and taking a constraint-oriented approach to

modelling the collaborative design process, a number of advantages can be obtained as follows:

- Design constraints are essentially a descriptive, rather than a prescriptive specification of design requirements, and thus they do not restrict how the product is to be designed, but instead, state what relationships must be satisfied for a set of design variables.

- Research work in the Artificial Intelligence field has provided a number of methods for representing constraints and reasoning about constraint violations [Bowen and Bahler, 1992; Serrano and Gossard, 1988]. A set of algorithms have been introduced to deal with backtracking constraint propagation [Dechter, 1992].

- The collaborative design process can be more efficiently and effectively managed and coordinated because, in product design, most of the design requirements can be represented as computational design constraints.

- According to the axiomatic design theory, the function-form relation represented by design constraints can be analysed and categorised into three forms: uncoupled design, decoupled design, and coupled design. Then, decomposition techniques can be applied to reduce the coupling among sub-assemblies and parts maximally.

- Multiple perspectives of design constraints can be provided by partitioning and grouping design variables involved to facilitate designers in the design problem solving process. It helps designers understand the design rationale behind a design decision made by other designers who may be geographically dispersed.

In the following sections, the key concepts and the representation framework of the CDPM are delineated in detail. As the CDPM is proposed to work jointly with the CPDM presented in Chapter 4, the relationship between the two is also described.

5.3.2 Formal Definition

To facilitate the discussion of the proposed CDPM, some underlying concepts relevant to representing a collaborative design process are explained first in this section.

The overall *design objective* describes the specification of the desired outcome of a product design project and it may consist of a number of sub design objectives. A design objective may be defined in terms of a function to be achieved and can be represented as a number of *function requirement*s. A design objective can introduce some sub design objectives, and by this way a design objective tree is created. Correspondingly, a function requirement (FR) tree is built as shown in Figure 5.1. On level z, as for the name of every function requirement, the amount of the digits following the 'FR' denotes the decomposition depth of the function requirement tree, i.e. z.

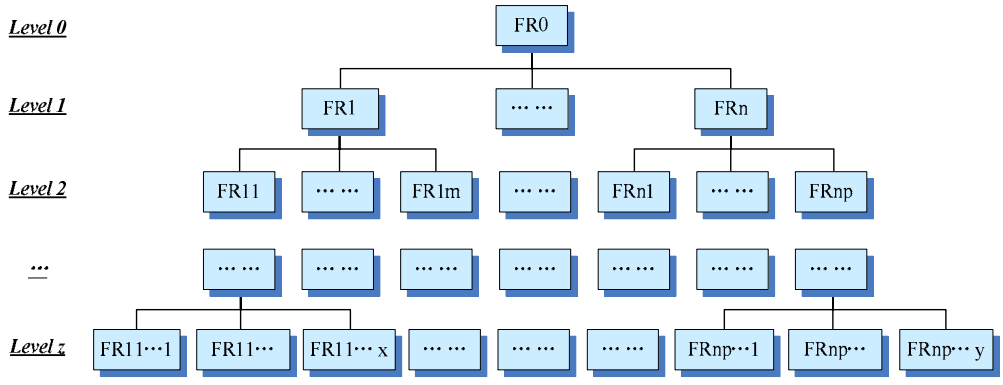


Figure 5.1: The function requirement (FR) tree

A *design product* is the product currently under design in order to meet all the design objectives. Corresponding to the product data module of the CPDM

which is proposed in Chapter 4, the design product is an assembly and comprises a hierarchy of sub assemblies and parts.

All the sub assemblies and parts of the design product are associated with a set of *design variable*s. All design variables comprise a design variable set X = $\{X_1, \ldots, X_n\}$ and they are associated with a set of domains D = $\{D_1, \ldots, D_n\}$ in which each domain is a set containing the possible values for the corresponding design variable. There are two types of design variables: abstract design variables and concrete design variables. An *abstract design variable* corresponds to a design variable defined at the sub assembly level while a *concrete design variable* corresponds to a design variable defined at the part level using parameters of features that define parts of the product. Thus, all abstract design variables can be refined by the combination of concrete design variables.

A *designer* is a human being capable of making design decisions. Parts of the design product, design variables, design constraints are associated with designers through ownership or interest relationships.

The *design activity* is carried out by a designer to achieve a design objective, i.e. satisfying a function requirement, by making a design decision to introduce a new sub assembly/part/feature and assign values to the corresponding concrete design variables, or delete an existing sub assembly/part/feature, or modify the values of concrete design variables of the existing feature. Figure 5.2 illustrates a mapping process among the function requirements and the design variables which define the physical form of the design product.

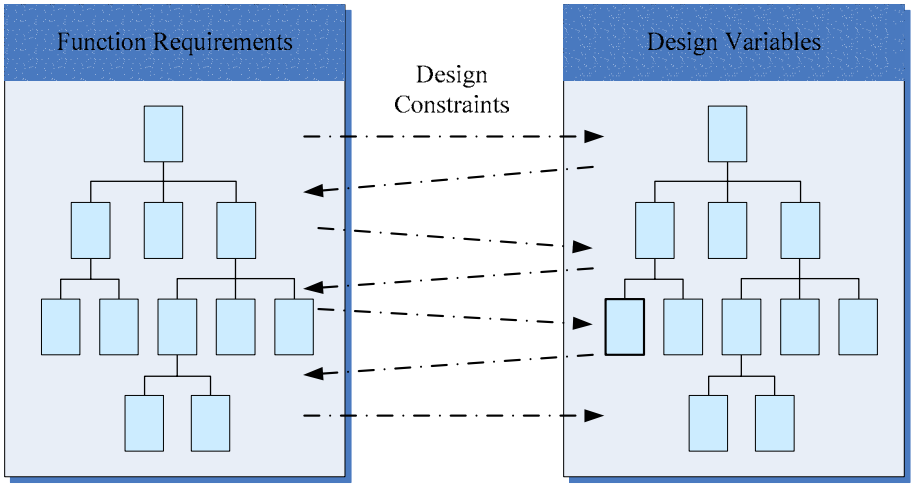


Figure 5.2: The mapping process among the function requirements and the design variables

A *design task* is associated with a number of design objectives. The whole design project can be viewed as a design task and it can be decomposed into many sub design tasks. A design task is carried out by executing a number of design activities to meet the associated design objectives.

A *design constraint*, defined based on a function requirement, is a limitation on the permissible values of a design variable within itself or with respect to other design variables. Each design constraint C is a relation R defined over a subset of design variables $\{ X_{i_1},\cdots,X_{i_{j(i)}} \}$ and $R(C_i)=R(X_{i_1},\cdots,X_{i_{j(i)}}) \subseteq D_i \times\cdots\times D_{i_{j(i)}}$. Corresponding to the distinction between abstract and concrete design variables, there is also a distinction between *abstract constraints* and *concrete constraints*. When the subset of design variables on which the design constraint is defined includes abstract design variables, this design constraint is named as an abstract design constraint. As the design process evolves, all the abstract design constraints are refined and represented by concrete design variables, thus evolving to be concrete design constraints.

The *design constraint network (DCN)* is a set of design constraints denoted by C = {C_1, …, C_m} which are interconnected by virtue of sharing design variables among various designers. The design constraint network can be used in design constraint propagation to monitor violations of design constraint. When a designer makes a design decision by assigning some values to a number of design variables, these values propagate through the design constraint network. Design constraints can be used to derive the values or value ranges of some design variables based on the values already assigned to some other design variables. They can also be used to verify whether the assignment of values of some design variables violates any design constraints after design constraint propagation.

A *design process state* is the design information known at a specific moment during a design process. This design information includes the current function requirements, design variables as well as their values, status of design constraints, etc.

A *design process* is a sequence of design activities and it can be represented by a sequence of design process state-transition pairs denoted by {< S_i , T_i>, i=1, …, n}, where S_i refers to the design process state at stage i, and T_i refers to the transition between S_i and S_{i+1} .

5.3.3 A Framework of Design Collaboration

In this research, a constraint-based Collaborative Design Process Model (CDPM) is employed to model the constantly changing product design process (Figure 5.3).

There are two main points in the proposed CDPM as follows:

1) Design constraints are formulated as a network of complex, dynamic, and hierarchical design constraints.

2) A collaborative design process is viewed as a sequence of state transitions from one design process state to another resulting from the design decisions made by participating designers. The precondition of a successful design process state transition is the consistency of all design constraints in the constraint network being preserved.

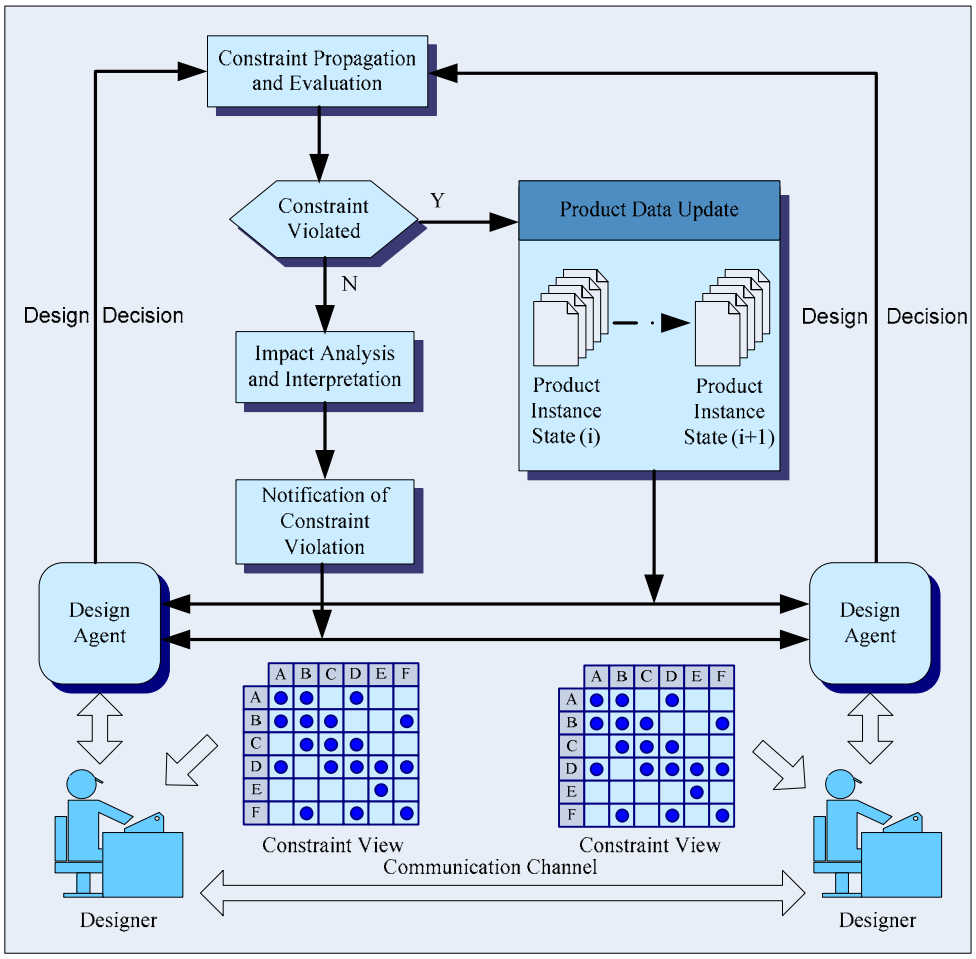


Figure 5.3: The constraint-based Collaborative Design Process Model (CDPM)

A product design project usually starts with the description of design objectives, and then follows with a decomposition of the overall design task into hierarchical sub design tasks. These design tasks each of which is associated with some design objectives are assigned to a number of designers.

A designer thus begins with the function requirements of the design task, and carries out design activities to meet the assigned design objectives.

In the collaborative design process, the efficient and effective coordination of design activities which are performed by various distributed designers concurrently is a key task in the domain of the application. The objective is to allow individual designers to make good design decisions and in the meantime to be notified of the effect of design decisions made by themselves or other designers. As mentioned before, the proposed CDPM is focused on facilitating the management and coordination of the collaborative design process. Figure 5.3 gives a description of this model.

There are two cases when a design constraint can be introduced. First, for each newly introduced design variable, there may be a design constraint on its value range. Second, when a design problem with a design objective is formulated, design constraints may be introduced among the design variables of the same sub assembly/part or different sub assemblies/parts of the design product.

Next, the design example introduced in Chapter 3 is taken to illustrate the coordination mechanism for a collaborative product design process provided by the CDPM as follows (the unit of design dimension is centimetre):

Step1: Suppose that at a certain stage, the design process state can be denoted as S_i and the detailed product data represented using the proposed CPDM is described in Section 4.4. Some design variables of the current product instance take the following values: DT.T1=4, DC.T2 = 1, and DC.H4 = 43.5.

Step 2: After Designer1 makes a design decision to assign a value of 60 to design variable DT.H2, the value of DT.H2 in the design constraint network is updated automatically to reflect the result of that design decision.

Step 3: A design constraint propagation event is triggered and at the same time the consistency of the design constraint network is then evaluated. In this case,

a design constraint, Constraint1, in the design coordination data module for the dining table is found to be violated.

Step 3: An impact analysis event is then triggered and the explanation of the constraint violation is also generated.

Step 4: A message stating the name and description of the violated design constraint (Constraint1) and the reasons (new value 60 for DT.H2, and current values for other involved design variables: DT.T1=4, DC.T2 = 1, and DC.H4 = 43.5) is then sent to those designers who are involved or interested in the violated design constraint (Designer1 and Designer2).

Step 5: As the consequence of the design constraint violation, Designer1 is required to negotiate with Designer2 and makes a redesign decision. Desiger1 assigns a value of 70 to the design variable DT.H2. This time, no design constraint is violated.

Step 6: A design process state transition from S_i to S_{i+1} is completed.

5.4 Summary

This chapter has given a detailed description of a constraint-based Collaborative Design Process Model (CDPM) which is proposed to facilitate the management and coordination of the collaborative design process. The main points are as follows:

- A formal and comprehensive method for modelling design process is of much interest to the development of computer supported collaborative design systems. A number of motivations for seeking such a method are examined.

- To facilitate the management and coordination of the collaborative design process and overcome the shortcomings of existing

methods for design process modelling, a constraint-based Collaborative Design Process Model (CDPM) is proposed and described in detail.

Chapter

6

Implementation Strategy of the Prototype System

6.1 Overview

In Section 1.1, the main requirements for a collaborative product design framework has been analysed and summarised. In Chapter 3, an agent-based system architecture for the proposed collaborative product design framework is designed and the functionalities of its components are described in detail. In this chapter, the implementation strategy and main technologies employed for the development of a prototype system of the proposed framework are presented in detail.

As described in Section 3.3, the agent-based framework supporting collaborative product design proposed in this research employs a distributed agent-based system architecture which is designed to be a network of software agents which interact with each other and participating designers to facilitate the collaborative design work.

The prototype system of the proposed agent-based framework is implemented on a network of computers with Microsoft Windows 2000/XP and Linux operating systems. A set of PolyCom® ViewStation FX® system and a number of PolyCom ViaVideo cameras are used for video conferencing purpose and they can work with the proposed design communication agent to facilitate real-time communications among human designers. The hardware environment for the development of the prototype system is shown in Figure 6.1.

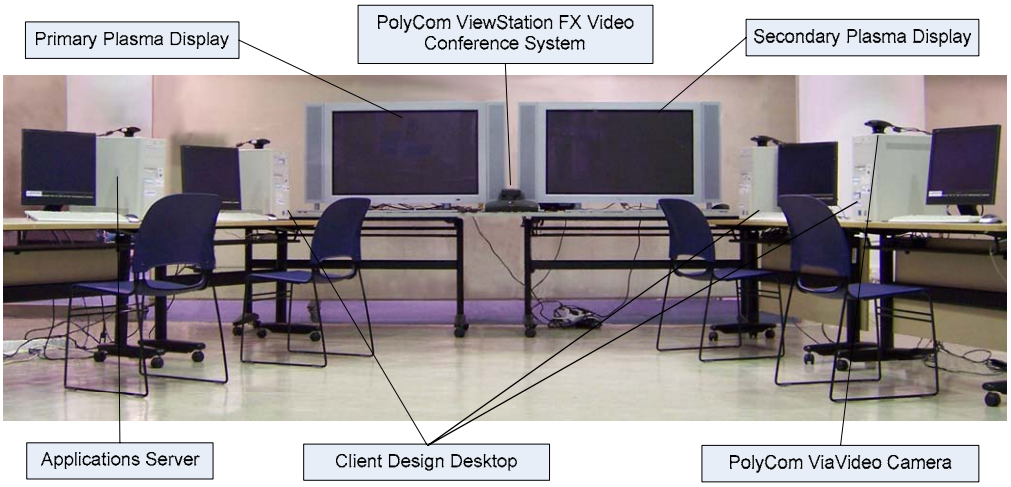


Figure 6.1: The hardware environment for the development of the prototype system

Java is chosen as the primary programming language for the system implementation. Visual C++ is also used as the programming language for the integration of legacy systems. CORBA (Common Object Request Broker Architecture) and JNI (Java Native Interface) are used to serve as the glue between Java-written applications and C++-wrapped native applications. JADE (Java Agent DEvelopment Framework) which is a software framework fully implemented in Java language is utilised to develop the agent-based framework. Currently, Autodesk Inventor® which is one of the most popular CAD systems and has an architecture similar to most of the 3D solid modelling design environments, is integrated into the design agents through

Autodesk Inventor 10.0 COM API. FIPA ACL is used to serve as the inter-agent communication language. The XML schema has been used to represent the collaborative product data model.

6.2 The Primary Programming Language

The Java is an object-oriented programming language developed by Sun Microsystems. The choice of the Java language as the primary programming language for the development of the proposed agent-based framework is based on many of its attributes that make it ideal for implementing intelligent agents and distributed multi-agent systems. These specific features include its object-oriented style, support for threads, distributed objects, architecture-neutral, network-centric design and they are discussed as follows. Bigus, et al. [1997] have written a book focusing on issues related to the development of intelligent agents using the Java language.

(1) Java supports the development of intelligent agent applications

Intelligent agents are generally characterised by a number of features including autonomy, intelligence, and mobility. The development of intelligent agent applications can benefit from various specific features of the Java language as follows:

Autonomy: In order to make an intelligent agent, which is a software program, to be autonomous, it must be a separate process of thread. Java supports threaded applications and Java applications run in separate processes and as such can be long running and autonomous. An agent programmed using Java can communicate with other programs using sockets and in an application it can be a separate thread of control. With the support of Java's event-processing mechanisms, agents are always waiting, ready to respond to a user request or a change in the environment, thus representing a certain degree of autonomy.

Intelligence: The intelligence of intelligent agents can range from hard-coded procedural or object-oriented logic to sophisticated reasoning and learning capabilities. Java provides various base functions required to support these behaviours.

Mobility: Java's portable bytecodes and JAR files allow groups of compiled Java classes to be sent over a network and then executed on the target machine. Java applets provide a mechanism for running Java code remotely via a web browser.

(2) Object-oriented language

Java is designed to be object-oriented and it provides a clean and efficient object-based development platform to implement distributed network-based systems. Programmers using Java can access existing libraries of tested objects that provide functionalities ranging from basic data types through I/O and network interfaces to graphical user interface toolkits. These libraries can be extended to provide new behaviours.

(3) Architecture Neutral

Java is designed to support applications that will be deployed into heterogeneous network environments. To accommodate the diversity of operating environments, the Java compiler generates bytecodes which is an architecture neutral intermediate format designed to transport code efficiently to multiple hardware and software platforms. The interpreted nature of Java permits Java programs to run on any platform that has a Java Virtual Machine available.

(4) Simple, Robust and Secure

Compared to another object-oriented programming language C++, Java removes various unnecessary complexities and becomes more robust. There

are no explicit programmer-defined pointer data types, no pointer arithmetic, and automatic garbage collection with a simple memory management model in Java. With security features designed into the language and run-time system, Java is very suitable to be used to build distributed systems with much security consideration.

6.3 The Agent Development Framework

To ease the realisation of agent applications, powerful frameworks and supporting standards, methodologies for facilitating the efficient development of agent-based systems are much needed.

The Foundation for Intelligent Physical Agents (FIPA)[12] is an IEEE Computer Society standards organization that promotes the interoperation of heterogeneous agents and the services that they can represent and a set of specifications which are commonly referred to as "the FIPA standard" have been released. The FIPA specifications can be grouped into five main categories: agent communication, agent transport, agent management, abstract architecture and applications.

So far, a large number of agent development frameworks have been proposed in the past few years, to name a few of those more well-known ones, JADE [JADE, 2006], JATLite [JATLite, 2003], AgentBuilder [Acronymics, 2004], RETSINA [Sycara, et al., 2003], Aglets [Aglets, 2002], JACK [JACK, 2006], ZEUS [Nwana, 1999], FIPA-OS [FIPA-OS, 2001], dMARS [Mark, et al., 1997], OOA [OOA, 2005]. Shen, et al. [2000] have provided a broader introduction to most of these agent development frameworks. Another extensive survey of agent construction tools can be found at http://www.agentbuilder.com/AgentTools/index.html.

[12] FIPA: The Foundation for Intelligent Physical Agents (FIPA) is a body for developing and setting computer software standards for heterogeneous and interacting agents and agent-based systems. Its official website is http://www.fipa.org.

Based on the results of the comparative evaluations of different agent development frameworks reported in some studies [Silaghi, 2005; Vrba, 2003; Oliveira, 2004; Perdikeas, et al., 1999], the JADE (Java Agent DEvelopment Framework) is selected as the agent development framework in this research according to the following considerations:

1) The JADE is developed to be in full compliance with the FIPA specifications.

2) The JADE simplifies the development of distributed agent-based applications by offering runtime and agent programming libraries, as well as a rich suite of graphical tools to support the debugging, management and monitoring phases of the agent life.

3) Agents realised with JADE are capable of controlling their own thread of execution and, therefore, they can be easily programmed to initiate the execution of actions without human intervention just on the basis of goal or state changes.

4) The JADE is an open-source project and it can be obtained without any charge. Furthermore, the user-driven development approach allows both users and developers to contribute with suggestions and new code.

The JADE is available at http://jade.tilab.com and its latest version is 3.4.1 (released on November 2006). A detailed description of the JADE can be found in [Bellifemine, at el., 2001, 2003, 2007]. The use of the JADE in this research can be described from two different points of view. On the one hand, the JADE works as a run-time platform for the proposed agent-based collaborative product design framework, supporting the application agents in terms of message passing, agent life-cycle managing, and so on. On the other

hand, it is a Java framework that is used to develop the application agents in the proposed framework using its agent programming libraries.

A JADE platform is itself a distributed system, since it can be split over several hosts. It comprises a number of *Agent Containers*, each living in a separate Java Virtual Machine and delivering run-time environment support to some application agents. A single special main agent container is always active in a JADE platform and all other normal agent containers register with it as soon as they start. Figure 6.2 shows the architecture of a JADE platform. Two mandatory system agents, the AMS (Agent Management System) and the default DF (Directory Facilitator), run within the main agent container. The AMS is an agent that exerts supervisory control over access to and use of the JADE platform and is responsible for authentication of resident application agents and control of agent registrations. In the proposed agent-based framework, the AMS works as the Agent Manager. The DF is also an agent which provides a yellow pages service by means of which an agent can find other agents providing the services it requires in order to achieve its goals.

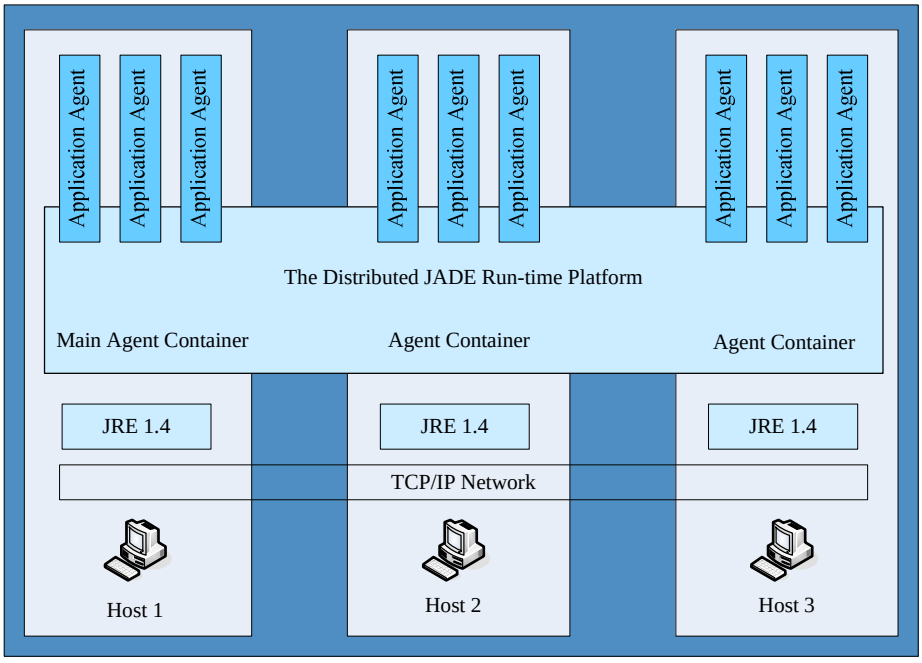


Figure 6.2: The JADE architecture

The JADE is selected to be the run-time platform for the proposed agent-based framework and it is also used to develop the proposed agents. Corresponding to the system architecture designed in the Chapter 3, the Agent Manager, the Project Management Agent, and the Design Coordination Agent are implemented and run in the main agent container of the JADE run-time platform on an application server. The implemented Product Data Management Agent runs in a separate agent container on a product data server. A Design Agent and a Design Communication Agent run in a separate agent container on a design client.

6.4 Internal Structure of the Software Agents

The software agents in the proposed agent-based framework are implemented using Java and the JADE agent programming libraries. The internal structure of the software agent is shown in Figure 6.3. Each software agent has a life-cycle manager, a scheduler of behaviours, an agent behaviour set, domain dependent agent resources, and a communication layer.

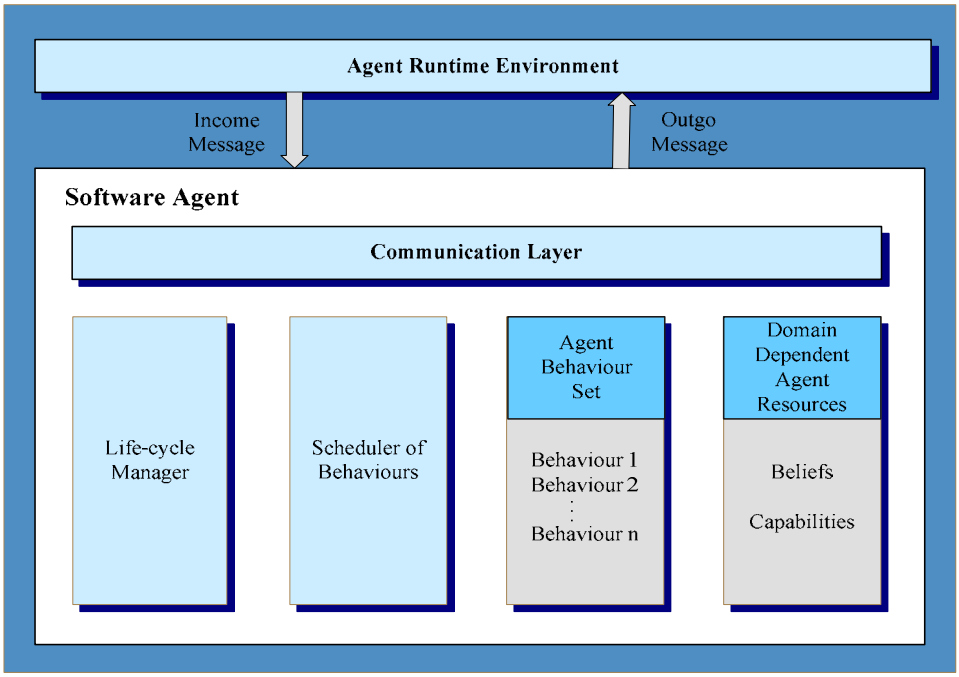


Figure 6.3: Internal structure of the software agents

The *life-cycle manager* is responsible for performing the life cycle operations (initializing, suspending, or terminating the software agent) on the software agent. The JADE agent classes are based on a method, called setup, that performs the agent initialisation, and another method, called takedown, that performs cleanup operations at the end of its execution.

The *agent behaviour set* includes a collection of behaviours which are scheduled and executed to fulfil the designed functionalities of the software agent. A behaviour represents a task that an agent can carry out. Agent behaviours can be composed of other behaviours and can be executed either a single time (one-shot behaviours) or different times (cyclic behaviours). Agent behaviours are based on a method, called action, which defines the operations to be performed when the behaviour is in execution.

The *scheduler of behaviours* is responsible for managing automatically the scheduling of agent behaviours. It carries out a round-robin non-preemptive scheduling policy[13] among all agent behaviours available in the task queue.

The *domain dependent agent resources* include the beliefs and capabilities of the agent in an application domain. Beliefs represent the informational state of the agent - in other words its beliefs about the application domain (including itself and other agents).

The *communication layer* provides a mechanism to facilitate the interaction with other software agents through the JADE runtime environment. Software agents communicate with each other through the FIPA-ACL (Agent

[13] A round-robin non-preemptive scheduling policy: Round-robin is one of the simplest scheduling algorithms for processes in an operating system, which assigns time slices to each process in equal portions and in order, handling all processes without priority. Non-preemptive is a style of computer multitasking in which the operating system never initiates a context switch from a running process to another process.

Communication Language)[14] and an ACL message is modelled as a Java object that can be exchanged between software agents.

As the JADE is compliant with the FIPA specifications and it is used to develop the software agents and also deliver run-time support for the implemented agent-based framework, the software agents implemented in this research are compliant to the FIPA agent model.

6.5 Integration Issues with Traditional CAD Platform

As described in Section 3.4, a type of design agent which integrates with traditional CAD software is proposed to facilitate design collaboration. Currently, Autodesk Inventor® is integrated into the design agent through the Autodesk Inventor 10.0 COM API.

The selection of Autodesk Inventor® as the traditional CAD software to be integrated into the design agent is based on a number of considerations as follows:

1) Autodesk Inventor® is the most popular 3D product solid modelling system and based on advanced parametric modelling techniques seen in products like SolidWorks® and Pro/ENGINEER®.

2) It is designed to fully utilise both 2D drawings and 3D modelling capabilities in an easy and intuitive design environment.

[14] FIPA Agent Communication Language: FIPA-ACL is the agent communication language standardized by the FIPA foundation since 1997. It defines a set of 4 primitive and 18 composite communicative acts, together with a formal meaning for each of them. Each communicative action is semantically defined by its feasibility precondition, which states the condition that must necessarily hold for this action to be performed, and its rational effect, which states the result expected by agents performing this action. For example, the Inform communicative action is used to tell an agent a fact. Its precondition states the author believes this fact and also believes the receiver does not already know about it. Its rational effect states that the receiver comes to believe this fact. Further details of the complete FIPA-ACL communicative action library can be found at http://www.fipa.org/specs/fipa00037.

3) It provides leading-edge functional design capabilities so that users can create design based on the function requirements of a product before they commit to complex model geometry.

4) It supports most of the major 3D CAD industry file formats including STEP, DWG, DXF, IGES, Pro/ENGINEER® files, etc.

5) It offers a rich set of development interface, Autodesk Inventor 10.0 COM API, through which it exposes a set of functionality in an object-oriented manner using technology from Microsoft called Automation [15] . Using these development interfaces, it can be customised to automate specialised workflows in some specific environments.

The Common Object Request Broker Architecture (CORBA) is a leading object technology for enabling application integration. CORBA is a standard defined by the Object Management Group (OMG) that enables software components written in multiple computer languages and running on multiple computers to interoperate. It provides the best technical solution for integrating information and it is open, robust, heterogeneous, multiplatform, and multi-vendor supported. Simply stated, CORBA allows applications to communicate with one another no matter where they are located or who has designed them. CORBA uses an interface definition language (IDL) to specify the interfaces that objects will present to the world. CORBA then specifies a "mapping" from IDL to a specific implementation language like C++ or Java. IDL provides operating system and programming language independent interfaces to all the services and components that reside on a CORBA bus. It

[15] Automation: It was formerly known as OLE Automation and introduced by Microsoft. It enables one application to manipulate objects implemented in another application, or to expose objects so they can be manipulated. An Automation server is an application (a type of COM server) that exposes its functionality through COM interfaces to other applications, called Automation clients. The exposure enables Automation clients to automate certain functions by directly accessing objects and using the services they provide.

allows clients and server objects written in different languages to interoperate across networks and operating systems. A detailed introduction of CORBA can be found in [Soley and Stone, 1995].

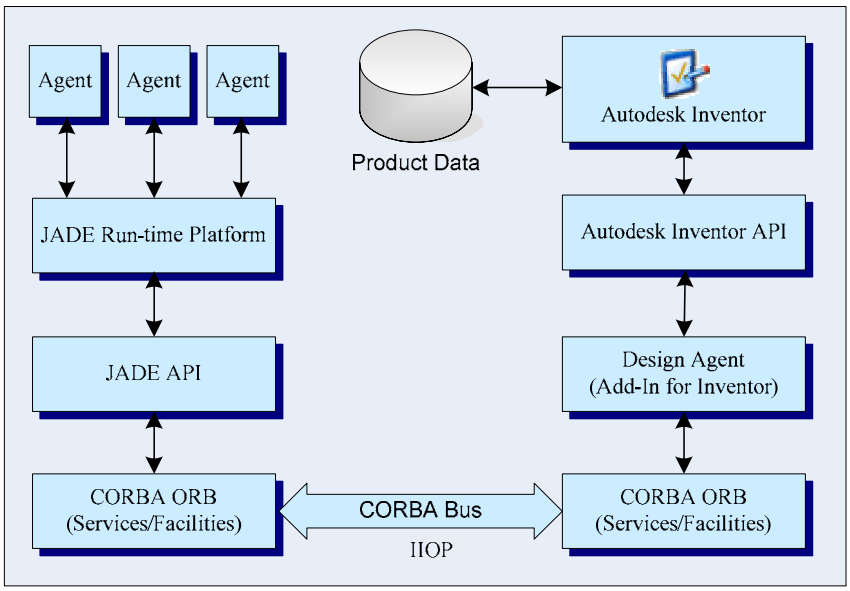


Figure 6.4: The integration of Autodesk Inventor with the design agent and the JADE platform

The integration of Autodesk Inventor with the design agent and the JADE platform is shown in Figure 6.4. In this research, for the implementation of the prototype agent-based framework, the design agent is implemented as an Add-In [16] for Autodesk Inventor using Visual C++ 6.0. The CORBA/IIOP [17] technology is used to realise the inter-platform communication between the design agent which is developed using Visual C++ and the Java-based JADE agent run-time platform.

[16] Add-In: An Add-In is a small program which runs in conjunction with another application that enhances the functionality of that program through the APIs of that program which expose a set of functionality of that application.

[17] IIOP: In distributed computing, GIOP (General Inter-ORB Protocol) is the abstract protocol by which Object Request Brokers (ORBs) communicate. Standards associated with the protocol are maintained by the Object Management Group (OMG). IIOP (Internet Inter-Orb Protocol) is the implementation of GIOP for TCP/IP. It is a concrete realization of the abstract GIOP definitions.

The Autodesk Inventor API is an object-oriented API. This means that a set of functionality within Autodesk Inventor is exposed as a set of "objects" through the API. A set of Autodesk Inventor objects can be queried and edited using the methods and properties of these objects. For example, a SketchArc object (an arc in a sketch), which supports properties like Radius, StartAngle and SweepAngle can be obtained, queried, and edited through the API.

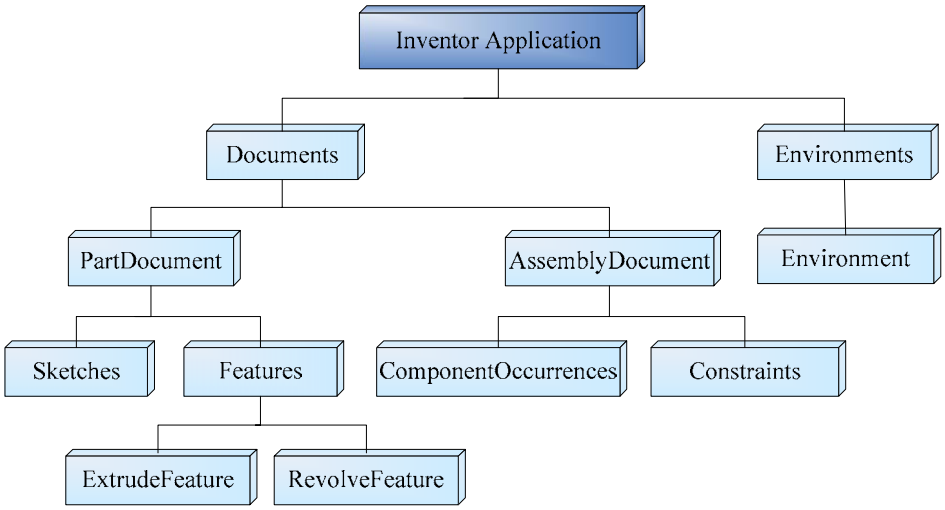


Figure 6.5: A simplified case of Autodesk Inventor's object hierarchy

The structure of the Autodesk Inventor API can be illustrated using a simplified case of Autodesk Inventor's object hierarchy as shown in Figure 6.5. The hierarchy shows the connections between various objects and describes how to traverse through the hierarchy to gain access to a particular type of object. The Application object is always the object at the top of the object hierarchy. By gaining access to the Application object, any other object in the API can then be accessed.

6.6 Summary

This chapter has given a detailed description of the implementation strategy and main technologies employed for the development of the prototype system. The main points are as follows:

- Java language is chosen to serve as the primary programming language for the development of the proposed agent-based framework due to many of its attributes that make it ideal for implementing intelligent agents and distributed multi-agent systems.

- After an investigation and a comparative analysis of several popular agent development frameworks, the JADE (Java Agent DEvelopment Framework) is selected as the agent development framework to implement the prototype system.

- As the most popular 3D parametric feature-based solid modelling CAD system, Autodesk Inventor is selected to be integrated into the design agent through Autodesk Inventor 10.0 COM API. The CORBA/IIOP technology is used to realise the inter-platform communication between the design agent and the JADE agent run-time platform.

In the next chapter, several real product design examples using the implemented prototype system are used to demonstrate how the proposed framework improves the efficiency and effectiveness of the management and coordination of a collaborative product design process. The main user interfaces of the developed prototype system are also described in detail.

Chapter

7

Validation and Discussion

7.1 Introduction

In previous chapters, the proposed agent-based collaborative design framework and the implementation details of the prototype system have been described. In this chapter, the results of the collaborative design experiments carried out using the developed prototype system are presented in detail.

The general goal of the design experiments described in this chapter is to validate the feasibility and applicability of the developed agent-based collaborative design framework using real design examples. The results from the experiments are used to address the problems of a complex collaborative design process identified in Section 1.1 of Chapter 1.

7.2 The Design Example

As described in previous chapters, the design of a set of dining table and chairs is used to validate the applicability of the proposed agent-based collaborative product design framework. In this chapter, this design example is

elaborated further through an experiment. In this experiment, a simulated design team is established to carry out the design tasks as a design project using the developed prototype system, recording the whole collaborative design process, and examining how designers can be assisted by the prototype system in managing and coordinating the collaborative product design process.

The product to be designed in this simulated project is a set of dining table and chairs. It is intended to seat six adults and it can also serve as a project table when required. A design team consisting of one design manager (Manager1: Kath Tang) and two designers (Designer1: Jason Wong and Designer2: Louis Lee) is set up to carry out this design project.

Real design would be much more complex than this simple simulation. The intention here is to demonstrate how the agent-based framework can support a design team, rather than a single user. The design example carried out by a design team including one design manager and two designers provides a scenario where potential conflicts may occur in a collaborative design process and the need for better communication and coordination becomes obvious.

7.2.1 Starting the System

To carry out collaborative design work using the developed agent-based prototype system, the JADE agent run-time platform and a number of software agents including the Project Management Agent, the Design Coordination Agent, and the Product Data Management Agent, must be kept running on the application server computer. If it is the first time to use the system, the JADE agent run-time platform needs to be launched first. Then, those previously mentioned software agents need to be created, and at this time they have registered themselves in the JADE agent run-time platform ready to provide designed support.

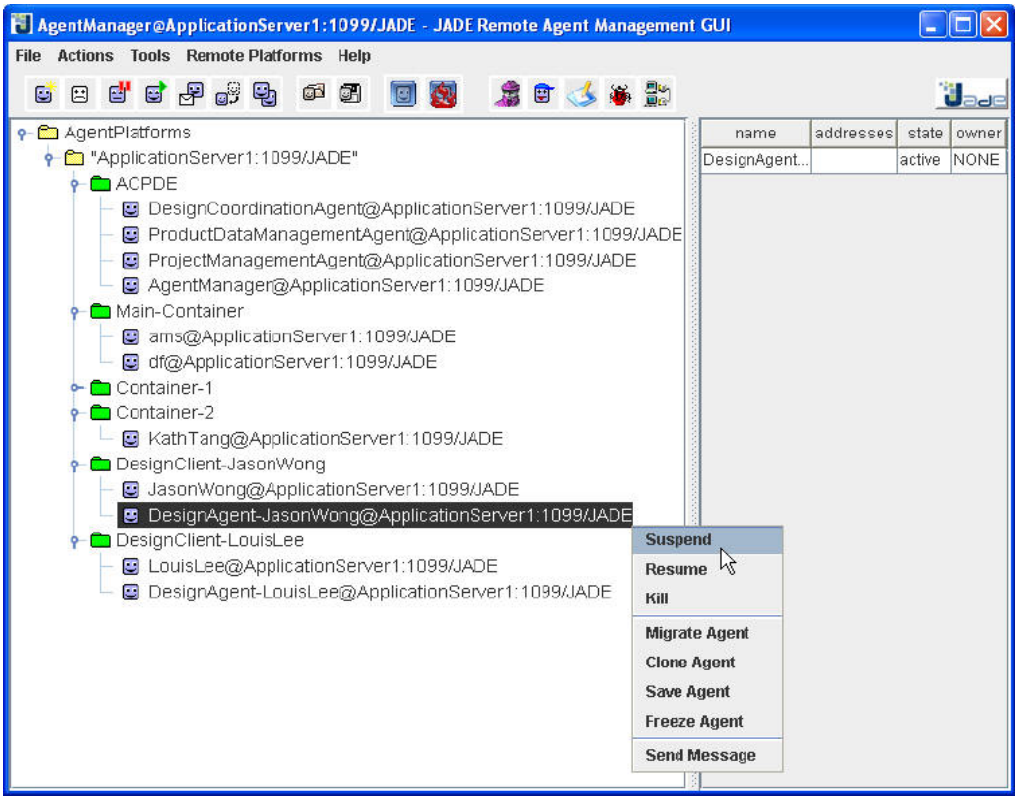


Figure 7.1: The main graphical user interface of the Agent Manager (JADE agent run-time environment) in which the Project Management Agent, the Product Data Management Agent, the Design Coordination Agent have already been created

The main graphical user interface of the Agent Manager (JADE agent run-time environment) is shown in Figure 7.1. There are a number of commands that can be executed from the menu bar (or the tool bar) of the Agent Manager to control the agent's life cycle, such as creating a new agent, terminating an existing agent, suspending or resuming an existing agent, etc. As shown in Figure 7.1, the Project Management Agent, the Product Data Management Agent, and the Design Coordination Agent are created and they run in the ACPDE agent container. In the main agent container, two more tool agents of the JADE, the AMS (Agent Management Service[18]) and the DF (Directory

[18] Agent Management Service: According to the FIPA architecture, this is the agent that is responsible for managing the agent run-time platform and providing the white-pages service.

Facilitator[19]) are created while the JADE is launched. They provide the management service to the platform and the yellow-pages[20] service respectively.

When the JADE agent run-time platform is running and the Project Management Agent, the Product Data Management Agent, and the Design Coordination Agent are created, a designer who is working with any design client having a network connection can create a Design Agent and a Design Communication Agent and have them join the running JADE agent run-time platform. As shown in Figure 7.1, two designers, Jason Wong and Louis Lee, working on two design clients have created two Design Communication Agents, whose nicknames are JasonWong and LouisLee, and two Design Agents, whose nicknames are DesignAgent-JasonWong and DesignAgent-LouisLee and these four agents have joined the JADE agent run-time platform (ApplicationServer1:1099/JADE).

7.2.2 Project Planning and Scheduling

The main graphical user interface of the Project Management Agent is shown in Figure 7.2. By interacting with the Project Management Agent, a project manager can describe the design specifications, decompose a complex design project into sub tasks, assign these tasks to designers, schedule these tasks, and track the whole design process. More importantly, the Project Management Agent will remind the project manager and the involved designers of the outstanding issues related to schedule execution. For the example design project, the design manager (Manager1) is responsible for planning, and scheduling the design project, and overseeing its schedule execution progress.

[19] Directory Facilitator: According to the FIPA architecture, this is the agent that provides the yellow-pages service.

[20] Yellow pages: It generally refers to a telephone directory for businesses, categorized according to the product or service provided. According to the FIPA specifications, a yellow-pages service allows agents to publish one or more services they provide so that other agents can find and successively exploit them.

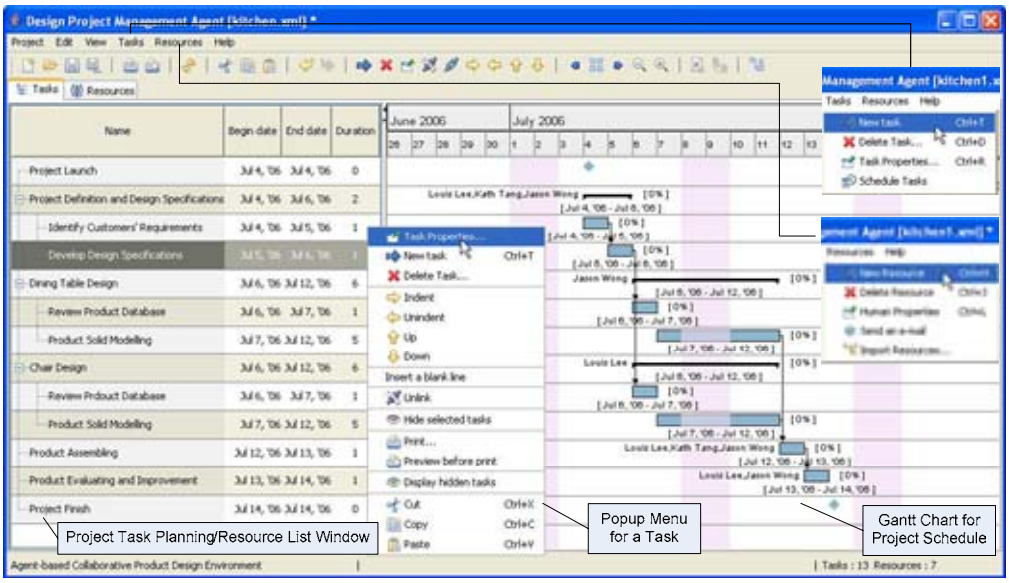


Figure 7.2: The main graphical user interface for Project Management Agent

To plan and schedule the example project, Manager1 creates a list of five major tasks of the project that the design team needs to complete for the project using the New Task button from the toolbar or choosing New Task from the Tasks menu. These five major tasks are as follows:

(1) Define the objective and detailed design specifications for the project.

(2) Design the dinning table.

(3) Design the chairs.

(4) Assemble the dinning table and chairs.

(5) Evaluate the whole design and make improvements.

Then Manager1 refines the design task list further by creating subtasks for task (1), (2), and (3) respectively. The final created design task list is shown in the Project Task Planning Window of Figure 7.2. Using the General tab of the properties dialog box shown in Figure 7.3, a number of general properties of the design task including task name, progress, priority, duration, begin date, end date, etc. are specified or changed.

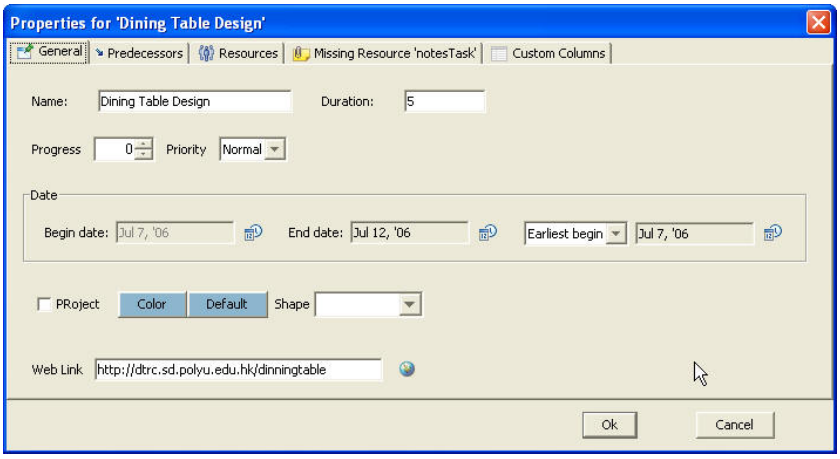


Figure 7.3: Specify the properties of a task using the task properties dialog box

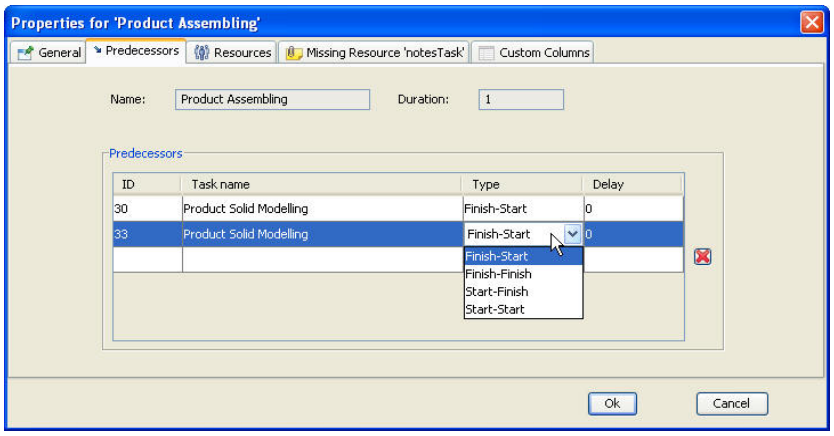


Figure 7.4: Define dependency relationships between tasks using the task properties dialog box

Linking tasks is a way showing team member how tasks are related to each other. By linking tasks, it is easy to make the necessary adjustments to the schedule whenever there are changes that affect the start or completion of other tasks. For example, the task of dining table design can not be started if the task of design specifications development has not been completed. In the Project Management Agent, when tasks are linked, the task that must be started or completed first is called the predecessor and the task that depends on the predecessor is called the successor. Using the task's Properties dialog box shown in Figure 7.4, four types of relationships listed in Table 7.1 between tasks can be specified.

Table 7.1: Four types of dependency relationships between tasks

Relationship Type	Description
Finish-Start	The successor task cannot start until the predecessor task finishes.
Start-Start	The successor task cannot start until the predecessor task starts.
Finish-Finish	The successor task cannot finish until the predecessor task finishs.
Start-Finish	The successor task cannot finish until the predecessor task starts.

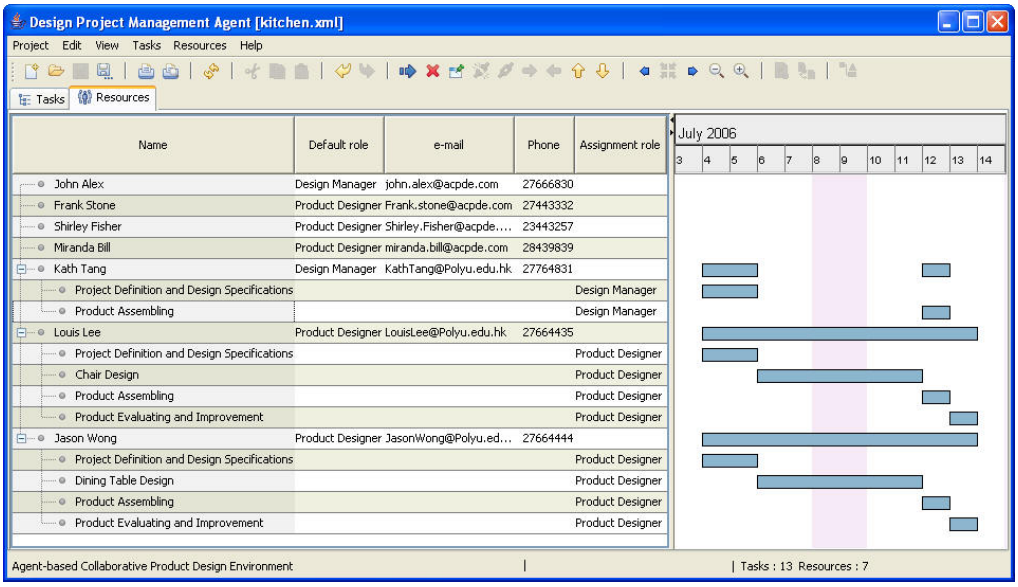


Figure 7.5: The resource list and the resource usage information

Resources for a design project are the individuals, facilities, equipments, and materials needed to complete the design project. A resource list is maintained by the Project Management Agent. Manager1 adds a new resource with a specified role to the resource list by using the New Resource button from the toolbar or choosing New Resource from the Resources menu. A resource list is shown in Figure 7.5. A resource can be assigned to a particular task using the Properties dialog box linked to that task. Individuals are a special type of resource called “human resources”. Human resources assigned to a task are scheduled to fulfil the task. As shown in Figure 7.6, Jason Wong is scheduled to carry out the design task “Dining table design”.

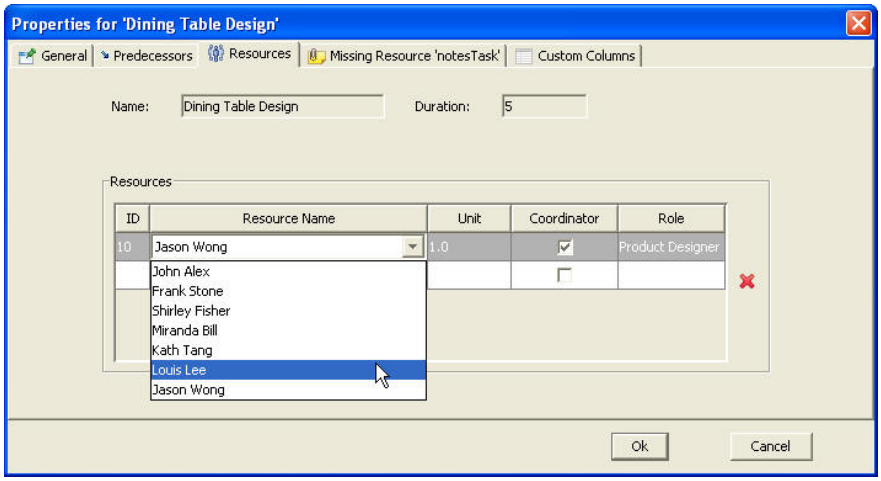


Figure 7.6: Assign resources to a task using the task Properties dialog box

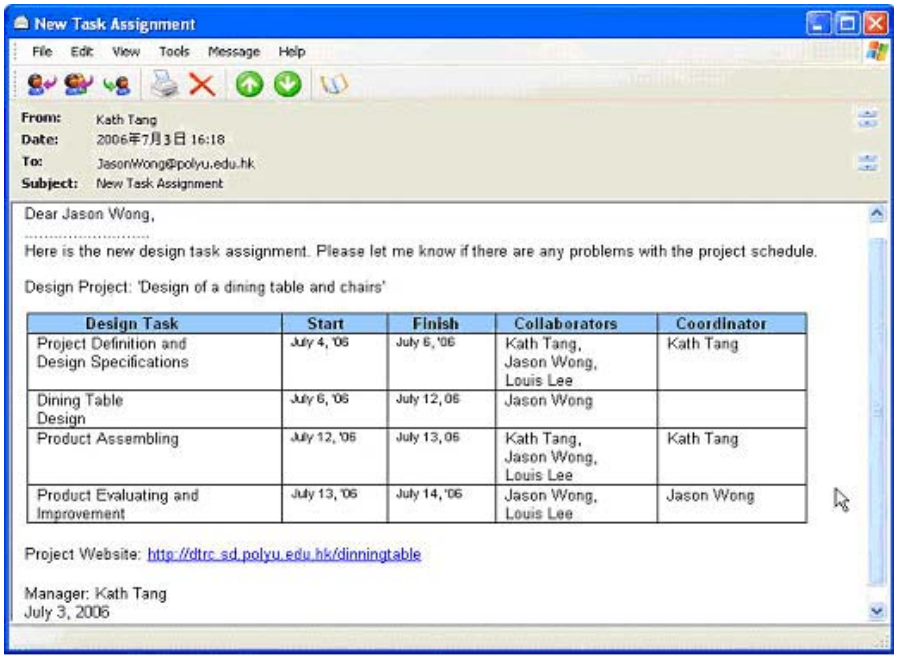


Figure 7.7: An email sent to Designer1 (Jason Wong) stating the task assignment generated by the Project Management Agent

After creating a task list for the design project, specifying the details for tasks, defining dependency relationships among them, and assigning required resources to the tasks, a design project is ready to be published as shown in Figure 7.2. By choosing Publish Task Assignments from the Project Menu, emails stating design task details are sent to assigned resources automatically. Figure 7.7 shows a task assignment email sent to Designer1 (Jason Wong).

7.2.3 Project Definition and Design Specifications

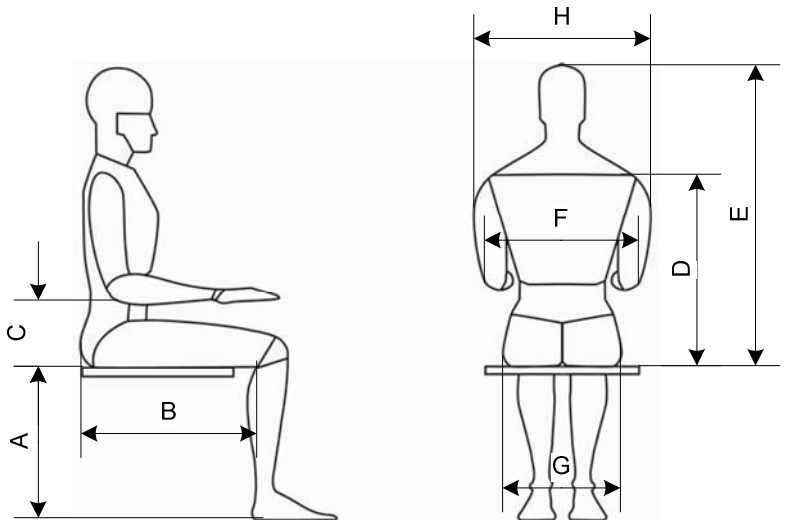
As scheduled by Manager1, the first project task is to work out the project definition and design specifications and it needs the collaborative work of the design team members.

To design a comfortable set of dining table and chairs, it is important to take into consideration how people use and interact with the product. Ergonomics (or Human Factors) is a separate discipline that studies the physical dimensions of people, the space that people move within, and objects and environment's psychological impact on people. The key anthropometric dimensions required for the design of a set of dining table and chairs are shown in Figure 7.8. In this example project, the goal is to accommodate people from the 5th percentile female to the 95th percentile male.

Based on the analysis of design problems and the above listed key anthropometric dimensions, a number of design specifications are identified by Manager1 (Kath Tang) after a message conference using Design Communication Agent with Designer1 (Jason Wong) and Designer2 (Louis Lee) as follows:

(1) Along the side of a rectangular table, a 60 cm minimum width per place setting or per side chair should be arranged for diners' convenience.

(2) To allow about a foot in front of each chair for a place setting, plus enough space between diners for food, dishes, etc. while not making conversation and food passing more difficult, the desirable width of the dining table should be between 76 cm to 102 cm.



Measurement	MEN				WOMEN			
	Percentile 5		95		Percentile 5		95	
	mm	in	mm	in	mm	in	mm	in
A: Popliteal Height	383	15.1	448	17.6	342	13.5	405	15.9
B: Buttock-Popliteal Length	421	16.6	494	19.4	401	15.8	469	18.5
C: Elbow Rest Height	228	9.0	298	11.7	215	8.5	284	11.2
D: Shoulder Height	557	21.9	641	25.2	518	20.4	594	23.4
E: Sitting Height Normal	858	33.8	958	37.7	809	31.9	891	35.0
F: Elbow-to-Elbow Breadth	371	14.6	489	19.3	348	13.7	478	18.8
G: Hip Breadth	295	11.6	355	14.0	310	12.2	382	15.0
H: Shoulder Breadth	398	15.7	469	18.5	363	14.3	438	17.2

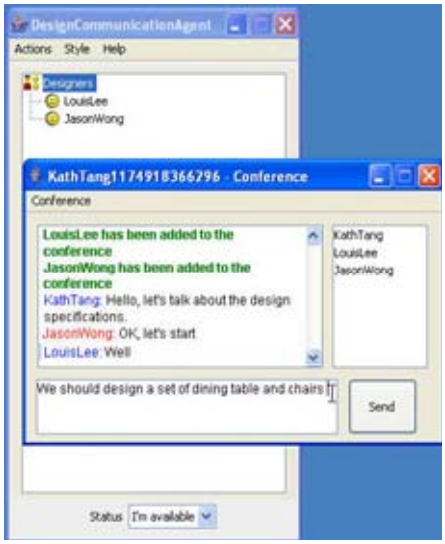
Figure 7.8: Key anthropometric dimensions required for the design of a set of dining table and chairs (The data is taken from Human dimensions of Chinese adults, GB 10000-88, A 95th percentile height would indicate that only 5% of the study population would have heights of greater dimension. A 5th percentile height would indicate that 95% of the study population would have heights of greater dimension.)

(3) For conventional right-angled sitting, the height of the chair seat should be no greater than the top of the diner's knee minus 5 cm.

(4) To dine while sitting on the chair, the vertical distance between the height of the top surface of the dining table and the height of the chair seat should be within a range of [20, 40] (cm).

(5) To save room when there is no person using them, the chairs should be able to be packed under the dining table.

(A) Main menu



（B) Multi-user message conference

(C) Online meeting

（D) Application sharing

Figure 7.9: Main graphical interfaces of the Design Communication Agent

7.2.4 Design Process Cooperating and Coordinating

After receiving design tasks, Designer1 and Designer2 start to design the dining table and the chair respectively. They, each working on a design client, carry out the design work using Autodesk Inventor® and cooperate and coordinate with each other with the support of a Design Communication Agent and a Design Agent.

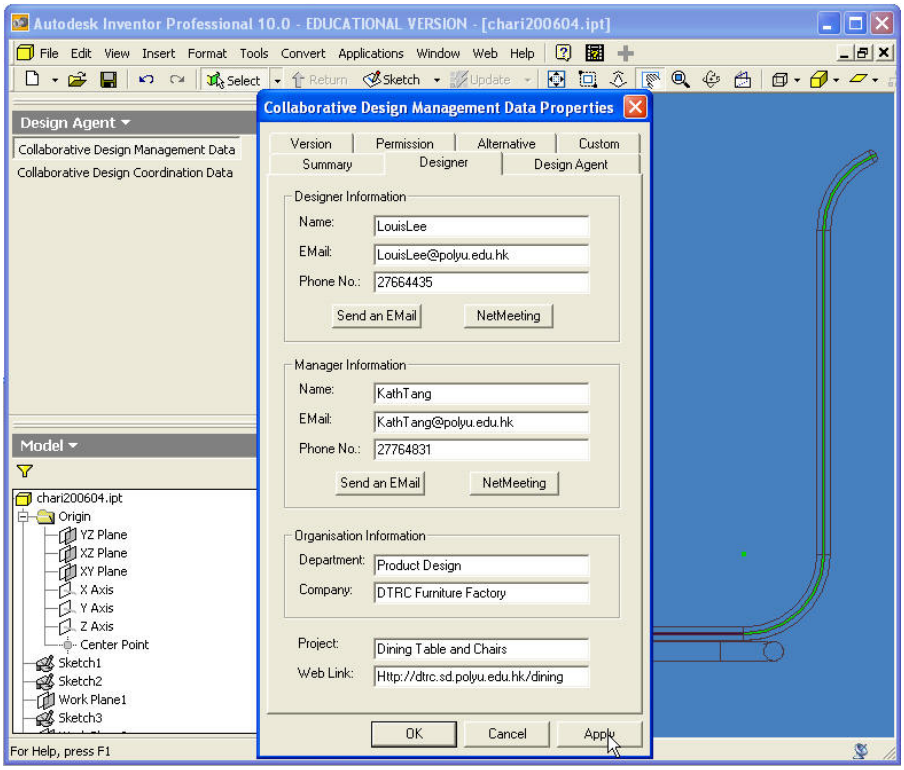


Figure 7.10: The Design Agent panel is used to specify the collaborative design management data

Various functions supporting design collaboration among distributed designers are provided by the Design Communication Agent that is created by a designer from the design client side. These functions include message chatting, message conference, online meeting, application sharing, whiteboard, and file transfer service. Figure 7.9 shows the main graphical user interfaces of the Design Communication Agent. A designer can exchange design ideas with another designer by just using the message chatting, or he/she can also invites

more than one designer to join a message or video conference to discuss design problems. Application sharing allows simultaneous co-modelling by sharing the CAD system control.

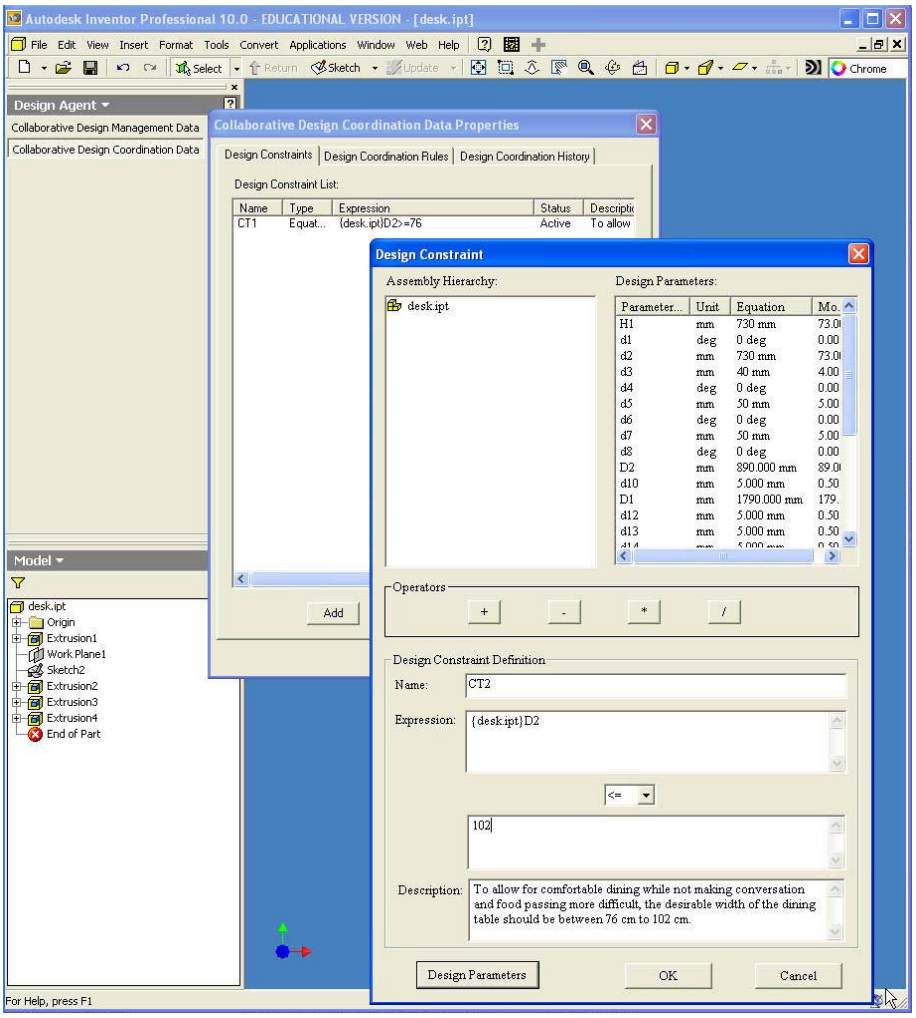


Figure 7.11: Using the Design Agent to specify the collaborative design Coordination data

Designer1 and Designer2 have set about developing 3D parametric solid models for the desired dining table and chair respectively. A Design Agent is created automatically when they launch the Autodesk Inventor® application on their design client and it joins the running JADE agent run-time environment automatically. Through the Design Agent, the collaborative design management data and the collaborative design coordination data can be specified and they are stored and maintained using the proposed Collaborative

Product Data Model (CPDM) described in the Chapter 4. Figure 7.10 shows the Design Agent panel and the graphical user interface to specify the collaborative design management data. Figure 7.11 shows the graphical user interface to specify the collaborative design coordination data.

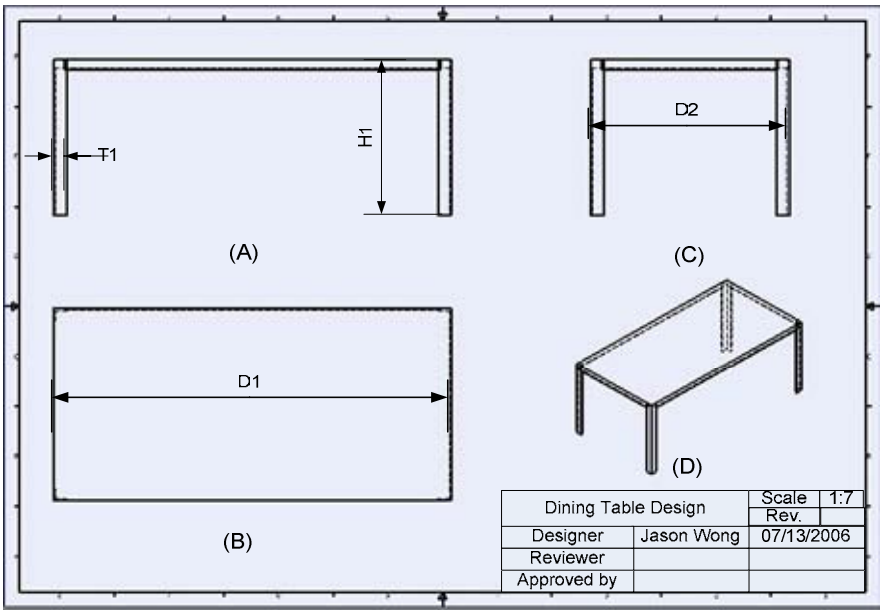


Figure 7.12: Drawing of a dining table with selected design parameters

Given the design objectives and specifications, Designer1 comes up with an initial design of the dining table based on his past experience. The drawing of the dining table and some of the design parameters associated with this design are shown in Figure 7.12 and Figure 7.13.

As described in the Chapter 5, in the proposed constraint-based Collaborative Design Process Model (CDPM), most of the design specifications can be represented as design constraints each of which is a relation defined over some design variables (parameters). A collaborative design process is viewed as a sequence of state transitions from one design process state to another resulting from the design decisions made by participating designers and the precondition of a successful design process state transition is the consistency of all design constraints being preserved.

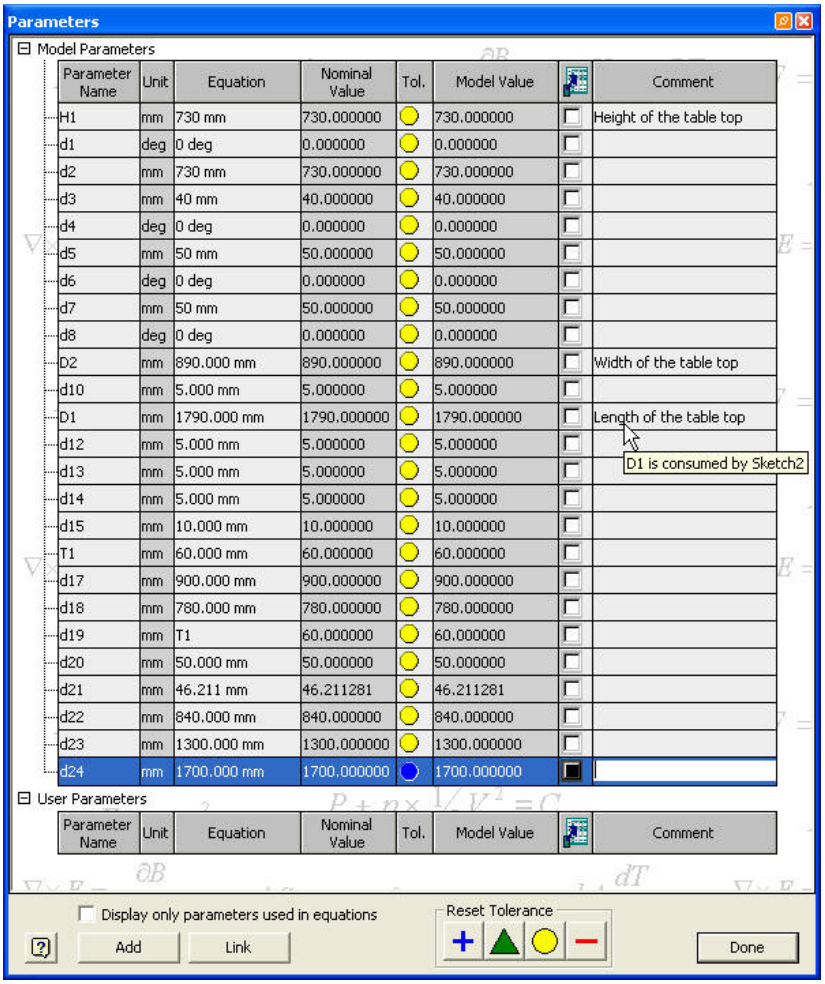


Figure 7.13: Design parameters for the dining table

There are two types of design constraints that can be defined. The first type of design constraints are defined on the part level and design variables involved in the definition of a design constraint of this type belong to only a single part. The second type of design constraints are defined on the assembly/sub-assembly level and design variables involved in the definition of a design constraint of this type belong to two or more parts or sub-assemblies.

Figure 7.11 shows the snapshot of the definition of a design constraint of the first type and Designer1 defines a design constraint CT2 which specifies that the width of the dining table top should be less than or equal to 102cm in order to ensure that the design of the dining table satisfy the design specification (2) identified in Section 7.2.3.

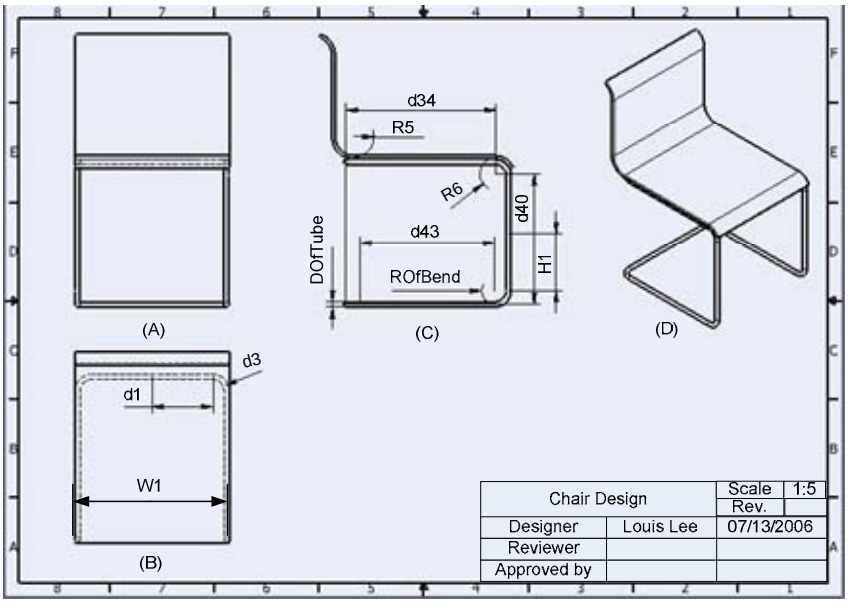


Figure 7.14: Drawing of a chair with selected design parameters

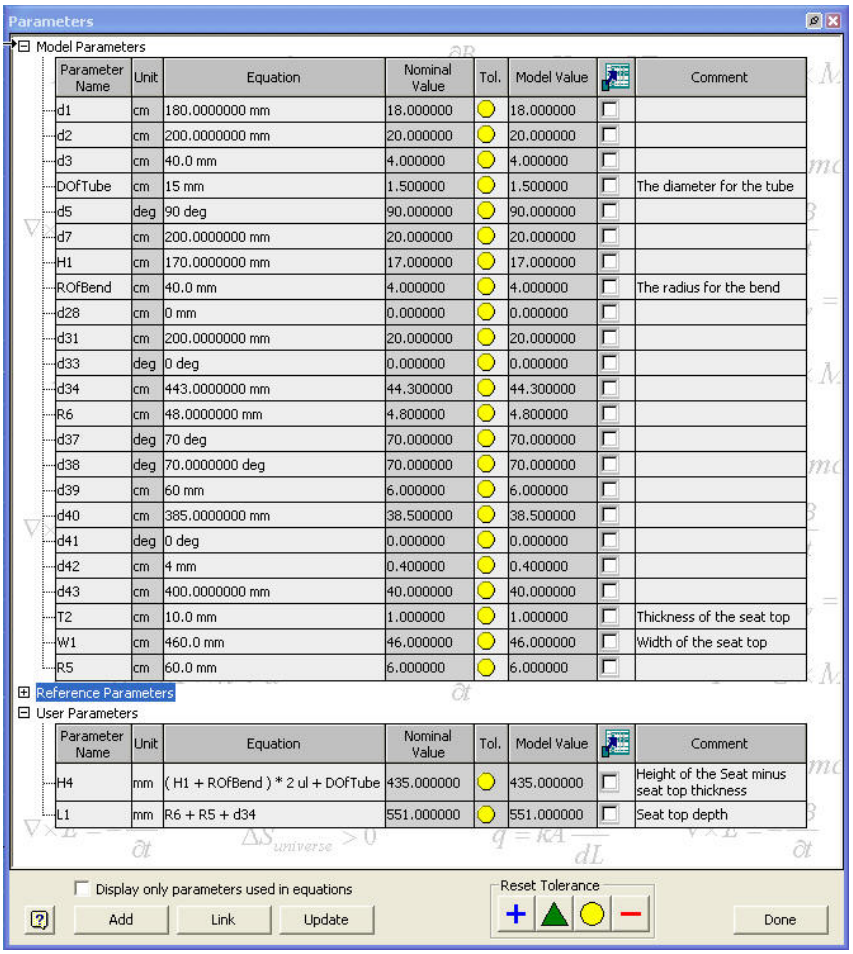


Figure 7.15: Design parameters for the chair

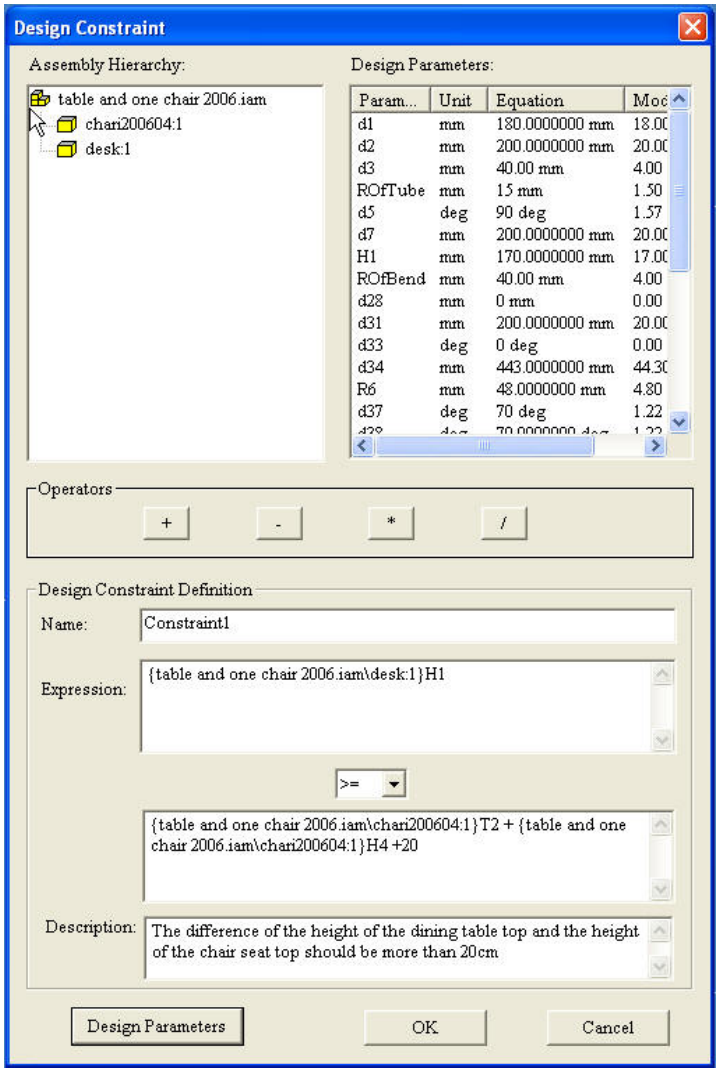


Figure 7.16: Defining a design constraint on the assembly/sub-assembly level

To specify a design constraint on the assembly or sub-assembly level, an initial assembly must be created first without considering the assembly constraints among the sub-assemblies and parts that comprise the product assembly. The drawing of the chair and some of the design parameters associated with this design are shown in Figure 7.14 and Figure 7.15. To satisfy design specifications (4) and (5) identified in Section 7.2.3, after exchanging design ideas with Designer2 using the Design Communication Agent, Designer1 creates an initial dining table and chair assembly. Figure 7.16 shows the snapshot of the definition of a design constraint on the assembly level and Designer1 defines a design constraint, Constraint1, which

specifies that the vertical distance between the height of the top surface of the dining table and the height of the chair seat should be more than 20cm.

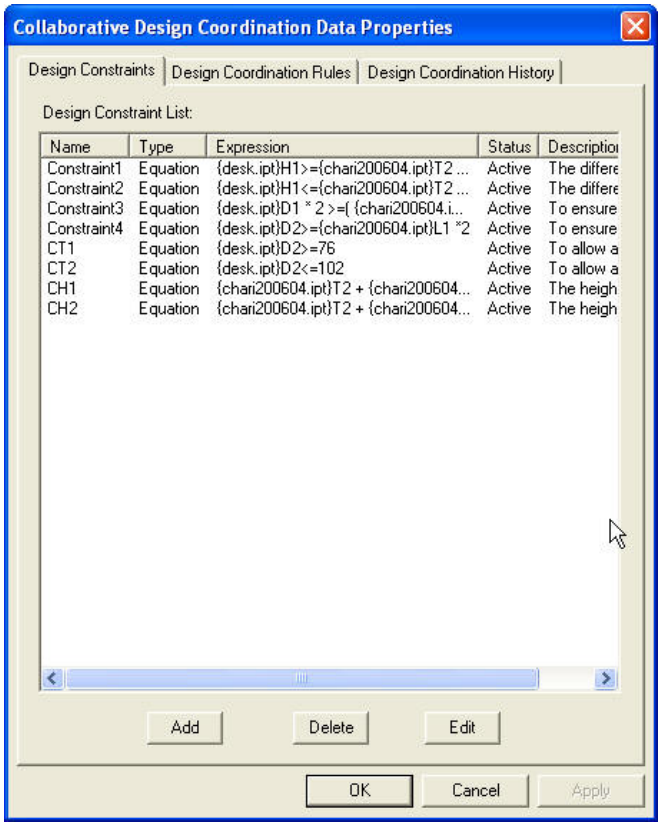


Figure 7.17: A list of design constraints defined for the dining table and chair assembly

Figure 7.17 shows a list of design constraints defined for the dining table and chair assembly. Eight design constraints are defined to satisfy design specifications (2), (3), (4), and (5) by Designer1 and Designer2. These design constraints form a design constraint network.

During the collaborative design process, a typical design coordination process is described as follows:

1) At a certain time, the design variables/parameters of the current product instance state take the following values:

   The dining table instance: H1= 73cm.

The chair instance: DOfTube=1.5cm, T2=1cm, ROfBend=4cm.

Figure 7.18: The message showing a design constraint is violated and its reason

2) As Designer2 is working on the detail design of the chair, a design decision is made by him and this design decision is to assign a value of 22cm to H1.

3) The design agent requests the Design Coordination Agent to check the consistency of design constraints against the design constraint network.

4) The chair is found to be referenced by a dining table and chair assembly. Then, the design constraints defined for the assembly are evaluated to check their consistency.

5) A design constraint (Constraint1) is found to be violated. Then, a message box indicating the design constraint violation and its reason is shown in Figure 7.18.

6) Designer2 can choose to redesign the chair or choose to inform other designers involved. If Designer2 chooses the later, the Design Agent then requests the Design Coordination Agent to perform design constraint propagation. Designer1 is found to be involved in the violated design constraint (Constraint1) according to the collaborative design management information for the design of the dining table.

Then, the Design Coordination Agent generates a message indicating the design constraint violation and its reason and sends it to Designer1 using the Design Communication Agent.

7) After the negotiation with Designer1 using the Design Communication Agent, Designer2 makes another design decision by assigning a new value of 17cm to H1. This time, no design constraint violation is found by the Design Coordination Agent.

8) The product data state is updated to reflect the design decision made by Designer2 by the Product Data Management Agent.

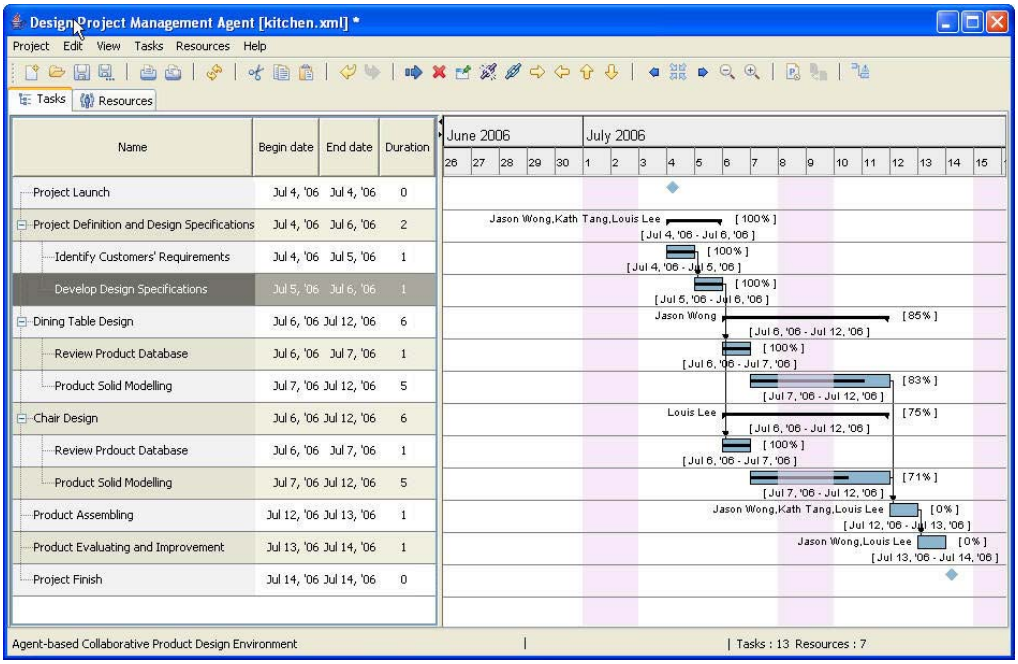


Figure 7.19: The updated design project progress

7.2.5 Collaborative Design Process Tracking

During the collaborative design process, the design manager communicates frequently with the design team members about the status of their design tasks, including percent complete, actual duration and remaining duration, using the Design Communication Agent. In this way, he/she keeps the design project

progress updated. If it runs behind schedule, or it runs ahead of schedule, he/she must make the necessary adjustments to the project plan to get the project back on track. Figure 7.19 shows the updated design project progress at a certain time.

7.2.6 Final Design Result

When all design subtasks of designing parts/sub-assemblies are finished, they are assembled and checked according to the relations and constraints with respect to the design specification and assembly requirements. In this design example, the design manager, Manager1, is responsible for the product assembling. He cooperates with Designer1 and Designer2 using the Design Communication Agent, especially through the application sharing tool, to carry out the product assembling work. The final design of a set of dining table and chairs which meets all the design specifications is shown in Figure 7.20.

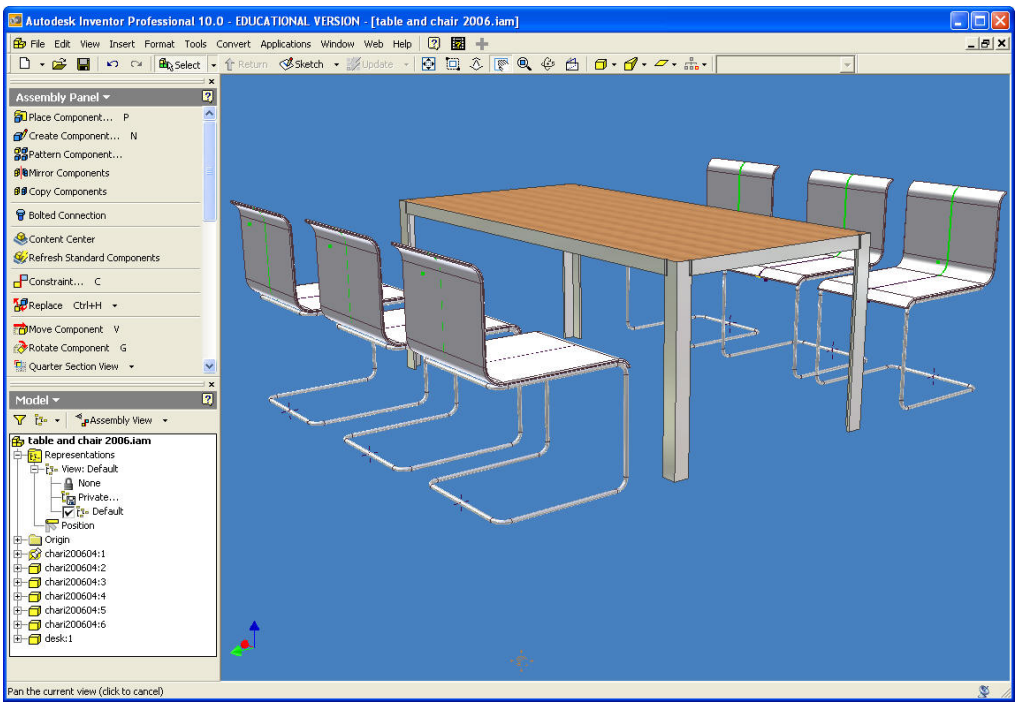


Figure 7.20: Final design of a set of dining table and chairs

7.3 Evaluation and Discussions

Through this application, it is demonstrated that the proposed collaborative design framework which adopts an agent-based approach and relocates managers, designers, systems in a unified knowledge representation schema is effective and capable of supporting design collaboration in terms of design management and design coordination.

The application example validated the effectiveness of the proposed Collaborative Product Data Model (CPDM) in representing collaborative design knowledge and sharing it among distributed designers.

The feasibility of the proposed constraint-based Collaborative Design Process Model (CDPM) in coordinating collaborative design processes and facilitating designers in making right decisions is established through the collaborative design process management and coordination.

It is demonstrated that traditional CAX software tools can be integrated seamlessly into a type of design agent to support design collaboration in an agent-based collaborative design framework.

The design example requires effective and efficient management and coordination works for cooperative design activities at multiple levels of the collaborative design process, although real design tasks can be much more complex than this. Nevertheless, the tested example includes basic features of real product design which requires supports for design communication, constraint satisfaction, 3D modelling and visualization in an integrated and user friendly manner. The collaborative product data model and collaborative design process model provides a strong basis for such integration. By extending the ability of design agents in a domain oriented way, it is possible to use the proposed agent-based collaborative design framework for larger scale design projects which may involve more designers and users. In this

simulated example, some tasks are carried out by human managers or designers in order to simplify the demonstration.

The implemented agent-based collaborative design prototype system has limited facilities for creating design agents which requires extensive design knowledge to accomplish. The implemented prototype system focuses more on enabling an effective communication process which is often more important in a collaboration than actually carrying out a specific design task requiring domain knowledge and experience of the designers. However, the objective and design specifications of the design of a set of dining table and chairs are achieved by the developed system framework and actual implementation. The results of the design example show that the proposed agent-based collaborative product design framework is effectively established to provide support to the management and coordination of the collaborative product design process.

7.4 Summary

This chapter has given a detailed description of a real design example whose design objective is to design a set of dining table and chairs using the developed prototype system.

The objective and design specifications of the design of a set of dining table and chairs are analysed in detail. Such a design project requires effective and efficient management and coordination works for cooperative design activities at multiple levels carried out by different design team members in the collaborative design process.

The results of the design example show that the proposed agent-based collaborative product design framework is effective and efficient in facilitating the management and coordination of the collaborative product design process.

Part IV

Conclusions

Part four consists of a concluding chapter that identifies the contributions made by this research and briefly discusses possible directions for future work.

Chapter

8

Conclusions and Future Work

8.1 Research Conclusions

The research presented in this thesis investigates the requirements of collaborative product design and develops a new agent-based framework to support collaborative product development, aiming at improving the efficiency and effectiveness of the management and coordination of a collaborative product design process.

In order to remain competitive in a global market, manufacturers have adopted a geographically distributed approach to forming new design processes. As such, design activities are being performed by design team members in different places. These designers need to collaborate with each other effectively and efficiently in order to develop their products faster, more cost effective, and with better quality. However, the success of collaborative design is undermined by the difficulties in managing and coordinating the collaborative design process as well as the differences between heterogeneous system architectures and information structures.

Agent technology is a promising approach to analysing, designing, and implementing industrial distributed systems. The way in which intelligent software agents residing in a multi-agent framework interact and cooperate with one another to achieve a common goal is similar to the way that human designers collaborate with each other to carry out a product design project. Thus, the hypothesis of this research is that a collaborative product design framework implemented by taking an agent-based approach can be used to assist human designers or design teams effectively and efficiently in a collaborative product design process.

The research methodology taken for developing the agent-based collaborative product design framework is based on both findings from a review of technologies and systems relevant to collaborative product design and a case study. This methodology is useful to identify the evolving and emerging research issues and to develop an adequate solution based on agent technology.

The conclusions, which can be drawn from this research work, are as follows:

(1) The literature review has shown that the existing methods for modelling product data and product design process are not sufficient for collaborative design and suffer from various limitations in terms of collaborative design management and coordination. The agent based approach is a promising alternative to the traditional client-server paradigm for implementing collaborative product design systems. However, most of the systems that have so far been implemented are domain dependent. Lack of integration with legacy design tools undermined their usability and the application of agent technology in collaborative design process is still under proof-of-the-concept prototype development stage.

(2) Based on an investigation on a complex collaborative design process and the findings from the reviewed systems, the key research issues of

this work are identified and a computational model of collaborative product design management and coordination is proposed.

(3) A multi-agent system architecture is proposed and it is designed to be a network of software agents which interact with each other and with the participating team members to facilitate the collaborative design work. These software agents include Project Management Agent, Design Agents, Product Data Management Agent, Design Coordination Agent, Design Communication Agent, and Agent Manager.

(4) A Collaborative Product Data Model (CPDM) is developed to support the implementation of the proposed collaborative product design framework. It is established by extending the popular parametric feature-based product data model and including extra information required for design management and coordination.

(5) A constraint-based Collaborative Design Process Model (CDPM) is proposed to model the constantly changing product design process. The model views a collaborative design process as a sequence of state transitions from one design process state to another resulting from the design decisions made by participating designers. The precondition of a successful design process state transition is the consistency of all design constraints in the constraint network being preserved.

(6) Technologies including Java, JNI, CORBA, etc, are chosen to implement the agent-based prototype system. JADE is selected to serve as the run-time platform for the prototype system. These technologies are widely supported, as they are free of cost and popular among the research community. Autodesk Inventor® is integrated into the design agent through Autodesk Inventor 10.0 COM API. The integration of all these technologies achieved in this research results in a reliable, easy-

to-extend, distributed agent-based collaborative product design prototype system implementation.

(7) Design experiments are carried out to validate the feasibility and applicability of the developed agent-based collaborative design prototype system using a real design example. The findings of these experiments demonstrated that the proposed agent-based framework is capable of supporting design collaboration among geographically distributed design team members and has the following advantages:

- It improves the effective communication and cooperation among geographically distributed designers during the collaborative design process.
- It facilitates design managers in planning and scheduling the design project as well as tracking the whole collaborative design process to keep it progressing smoothly as scheduled.
- It improves design coordination by informing involved designers of any design constraint violation and its reason and helps individual designers make good design decisions.
- It improves the sharing of product data and design knowledge.
- It reduces product development time by eliminating most feedbacks and iterations caused by not sharing product data.

This study contributes to the field of computational design and computer supported collaborative work by forming an agent-based framework, in which a collaborative design process is supported by a new product data model and a new process model. This research takes advantage of new computational techniques such as knowledge based systems, intelligent control and constraint management. The components of an agent-based collaborative design

framework are implemented, tested and evaluated in the context of a real 3D product design example. The purpose of using this real design example is to provides a solid ground for studying the feasibility of the implementation, which, when abstracted or over simplified, would reduce the complexity required to observe and analyse the advantages of using agent-based technology. The prototype system implemented in this thesis, due to the scope of the study and the limited time, is still largely experimental with certain limitations. The implemented key components nevertheless have shown promising advantages of a fully developed prototype system, which requires future work and more time to develop.

8.2 Future Work

There are some areas where further research work could be carried out in order to enhance and enrich the support provided by the agent-based collaborative product design framework developed in this thesis.

8.2.1 Extension of the Collaborative Product Data Model

In this research, a Collaborative Product Data Model is proposed to facilitate efficient and effective management and coordination of a collaborative product design process. However, functional information of the product could be used to further extend this product data model. The functional information of the product includes the purposes that a designer wants to pursue, the functions that the product performs to fulfil the purposes, the design rationales that justify the design decisions the designer makes. The incorporation of functional information into the product data model can further improve the product sharing among distributed designers by helping designers understand the design rationales behind the design decisions. Furthermore, it helps designers to learn from past design experiences.

8.2.2 Extension of the Collaborative Design Process Model

The functionalities of the proposed constraint-based Collaborative Design Process Model in the implemented prototype system could be further enhanced in order to support the design coordination.

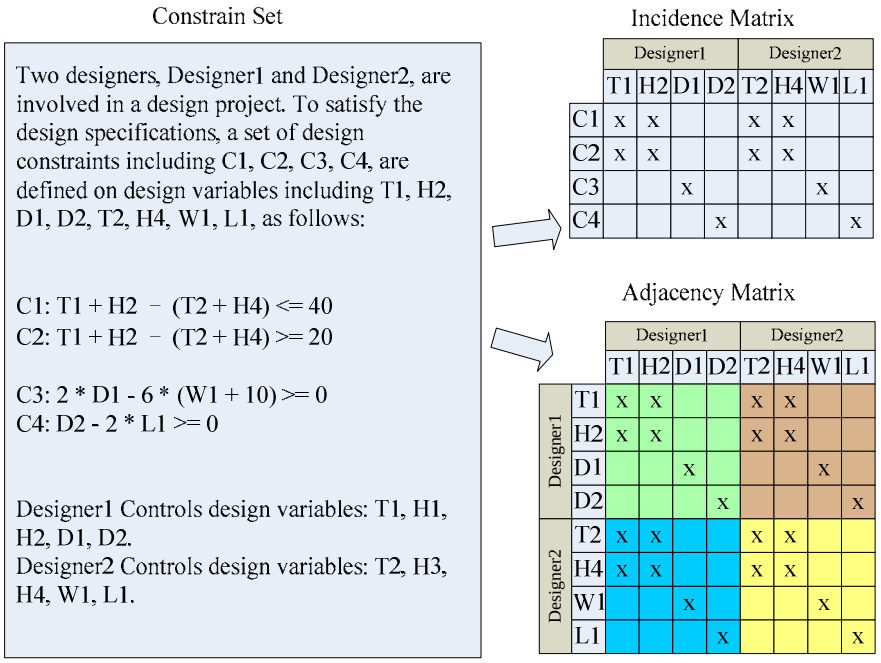


Figure 8.1: Various design constraint views in the form of matrix

Multiple perspectives of design constraints can be provided by partitioning and grouping design variables involved to facilitate designers in the design problem solving process as shown in Figure 8.1. In the form of incidence matrix, rows represent design constraints and columns represent design variables and each element in the matrix M is either empty or ‘x’. The ‘x’ symbol in element M_{ij} indicates that variable j appears in constraint i. In the form of adjacency matrix, both rows and columns represent design variables and each element in the matrix M is also either empty or ‘x’. The ‘x’ symbol in element M_{ij} indicates that variable j relates to variable i in more than one common design constraint. Through representing design constraints in the form of matrix, by partitioning and grouping design variables according to

different designers, the design team can obtain a clear insight regarding the relationships between design variables that he/she controls and those controlled by other designers. Furthermore, they can be aware of the possible impact of their design decisions on others' design.

8.2.3 Further Development of the Project Management Agent

For the time being, design project management activities including the decomposition of design tasks, planning and scheduling them, etc, are mainly carried out by human design managers interacting with the Project Management Agent according to their past experiences. However, functionalities supporting design managers to perform the above design project management activities could be developed to enhance the functionalities of the Project Management Agent.

The decomposition of design tasks could be facilitated by using a Design Structure Matrix (DSM) which is an efficient and commonly used method of showing the relationships between tasks within a project.

The planning and scheduling of design tasks could benefit from multi-agent distributed scheduling since the actual designers are humans who may be working simultaneously on different design projects. Each human designer could benefit from having an assistant agent that negotiates with others to come up with a design schedule for the designer and it is similar to the way in which meeting schedulers work.

8.2.4 Development of a Product Design Ontology

Ontology is concerned with the description of entities and the relations among entities of the world (or categories used to model the world), independently of any particular state or situation in which they happen to be. The development of an ontology for the product design domain can provide powerful

clarification and can be used to facilitate design knowledge sharing and improve design communication among design team members. For the proposed agent-based framework, the development of a product design ontology can facilitate the transfer of design knowledge among agents, agent learning, and interfacing agents and human designers.

8.2.5 More Experiments

Although the collaborative design experiments using the implemented prototype system demonstrate that the proposed agent-based collaborative product design framework is very powerful in supporting the management and coordination of a collaborative product design process. However, the design example is still very simple. In the future testing of the proposed framework, more designers will be involved in the collaborative design process with design examples of more substantial scale than the dining table and chairs presented.

8.2.6 Consideration on Human Factors

The study on designer's behaviour during the collaborative design process using the proposed agent-based collaborative product design framework needs to be considered. Human related problems could arise from interactions among designers as well as from the fact that the designers reside in different places, hence may have different cultures. These issues include expectations, conflict resolution, human interaction, etc.

In conclusion, collaborative design is a complex process and support for such a process cannot be provided by technologies alone. The agent-based system framework developed in this thesis offers the possibility of developing better coordinative and responsive systems which are based on the maintenance of consistent knowledge involved in a collaborative design process. The reality in design is, however, that more and more resources and efforts are being put in

reducing misunderstandings of intentions through fast and more coherent communication processes. The efficiency of collaborative design also largely involves organisational culture and IT strategies employed. The use of agent technology for more effective and intelligent design is a new trend, but the success of this technology still depends on how extensive design knowledge and skills can be enhanced by the employment of complex software agents.

This thesis has taken a systematic approach to this problem and has provided useful information and lessons. More studies and researches are needed to move this technology in the direction that helps designers do their work better and more innovatively without having to be exhausted by the long learning circles of various integrated design software systems. It is necessary to formulate and develop reliable design agents which perform certain design tasks without added complexity so that designers can concentrate more on resolving design issues that require human judgment thereby maximizing their potential in the design process.

Bibliography

Accreditation Board for Engineering and Technology (1985) Annual Report, October 1985.

Acronymics, Inc. (2004) AgentBuilder, http://www.agentbuilder.com.

Aderounmu, G. A. (2004) Performance Comparison of Remote Procedure Calling and Mobile Agent Approach to Control and Data Transfer in Distributed Computing Environment, Journal of Network and Computer Applications, 27:113-129.

Agent Platform Special Interest Group (2000) The Agent Technology: Green Paper, OMG Document agent/00-09-01, Version 1.0.

Aglets (2002) Aglets Software Development Kit, IBM Tokyo Research Laboratory (TRL), http://www.trl.ibm.com/aglets.

Alberts, L. K. (1994) YMIR: A Sharable Ontology for the Formal Representation of Engineering Design Knowledge, IFIP WG 5.2 Workshop on Formal Design Methods for CAD, Elsevier, pp. 3-32.

Alonoso, E. (2002) AI and Agents: state of the art, AI Magazine, American Association for Artificial Intelligence, 23(3):25-29.

Archer, L. B. (1984) Systematic Method for Designers, Cross N. (Eds.), Developments in Design Methodology, Wiley, and Chichester.

AS/NZS ISO 9004.1 (1994) Quality Management and Quality Subsystems – Part 1: Guidelines, Sydney, Standards Australia.

Bannon, L. J. (1992) Perspectives on CSCW: from HCI and CMC to CSCW, Proceedings of International Conference on Human Computer Interaction, August 1992.

Bellifemine, F. L., Caire, G. and Greenwood, D. (2007) Developing Multi-Agent Systems with JADE, John Wiley & Sons, Inc, USA.

Bellifemine, F. L., Caire, G. Poggi, A. and Rimassa, G. (2003) JADE: a White Paper, Telecom Italia EXP magazine, 3(3):6-19.

Bellifemine, F. L., Poggi, A. and Rimassa, G. (2001) Developing Multi-Agent Systems with a FIPA-compliant Agent Framework, Software-Practice and Experience, 31:103-128.

Bigus, J. P. and Bigus, J. (1997) Constructing Intelligent Agents with Java: A Programmer's Guide to Smarter Applications, John Wiley & Sons, Inc, USA.

Boothroyd, G. (1996) Design for Manufacture and Assembly: the Boothroyd-Dewhurst Experience, Huang, G. Q., (Eds.), Design for X: Concurrent Engineering Imperatives, Chapman & Hall, London, pp. 19-40.

Boothroyd, G., Dewhurst, P. and Knight, W. (2002) Product Design for Manufacture and Assembly, Second Edition. Marcel Dekker.

Bowen, J. and Bahler, D. (1992) Frames, quantification, perspectives, and negotiation in constraint networks for life-cycle engineering, Artificial Intelligence in Engineering, 7(4):119-226.

Bralla, J. G. (1996) Design For eXcellence, McGraw-Hill, Inc., New York.

Brazier, F. M. T., DuninKeplicz, B. M., Jennings, N. R. and Treur, J. (1995) Formal specification of multi-agent systems: a real world case, Proceedings of First International Conference on Multi-Agent Systems ICMAS'95, MIT Press, Cambridge, MA, pp. 25–32.

Brazier, F. M. T., Jonker, C. M., Treur, J. and Wijngaards, N. J. E. (2001) Compositional Design of a Generic Design Agent, Design Studies, 22(5):439-471.

Brown, B., Green, N. and Harper, R. (2001) Wireless World: Social and International Aspects of Wireless Technology, Springer Verlag, London.

Brown, D. C., Dunskus, B., Grecu, D. L. and Berker, I. (1995) SINE: Support for Single Function Agents, Applications of AI in Engineering X, 10:525-532.

Caldecote, V. (1986) Investment in New Product Development. Roy, R. and David W. (Eds.), Product Design and Technological Innovation, Open University Press, Milton Keyes, England, pp. 52-58.

Campbell, M. I., Cagan, J. and Kotovsky, K. (1999) A-Design: An Agent-Based Approach to Conceptual Design in a Dynamic Environment, Research in Engineering Design, 11(3):172-192.

Churchill, E. and Wakeford, N. (2001) Framing mobile collaborations and mobile technologies, Brown, B., Green, N. and Harper, R. (Eds.), Wireless world: Social and international aspects of the mobile age, Springer Verlag, London.

Cross, N. (1997) Descriptive Models of Creative Design: Application to an Example, Design Studies, 18:427-455.

Cross, N. (2000) Engineering Design Methods: Strategies for Product Design, Third Edition, John Wiley & Sons, Ltd., UK.

Cutkosky, M. R., Engelmore, R. S., Fikes, R. E., Genesereth, M. R., Gruber, T. R., Mark, W. S., Tenenbaum, J. M. and Weber, J. C. (1993) PACT: An Experiment in Integrating Concurrent Engineering Systems, IEEE Computer, January 1993, pp.28-37.

Cutkosky, M. R., Tenenbaum, J. M. and Glicksman, J. (1996) Madefast: Collaborative engineering over the Internet, Communications of the ACM, 39(9):78-87.

Dechter, R. (1992) From local to global consistency, Artificial Intelligence, 55: 87-107.

Dym, C. L. and Little, P. (2004) Engineering Design: A Project-Based Introduction, Second edition, John Wiley & Sons, Inc. New York.

Eiji, A., Kazuaki, I. (1992) Product modeling system in conceptual design of mechanical design, Robotics & Computer-Integrated Manufacturing, 9(4):327-334.

Erman, L. D. and Lesser, V. R. (1975) A Multi-Level Organization for Problem Solving Using Many Diverse Cooperating Sources of

Knowledge, Proceedings of 4th International Joint Conference on Artificial Intelligence, Tbilisi, ASSR, pp. 483-490.

Fang, W. D., Tang, M. X. and Frazer, J. H. (2003) Supporting Collaborative Product Design in an Agent Based Environment, Lecture Notes in Artificial Intelligence, 2718:447-460.

Fang, W. D., Tang, M. X. and Frazer, J. H. (2004) Constructing an intelligent collaborative design environment with distributed agents, Proceedings of the 8th International Conference on Computer Supported Cooperative Work in Design (CSCWD'2004), 1:329-335.

FIPA-OS (2001) FIPA-OS, http://sourceforge.net/projects/fipa-os.

Frazer, J. H. (2001) Design Workstation on the Future, Proceedings of the Fourth International Conference of Computer-Aided Industrial Design and Conceptual Design (CAID&CD 2001), International Academic Publishers, Beijing, pp. 17-23.

Fruchter, R., Reiner, K. A., Toye, G. and Leifer, L. J. (1996) Collaborative mechatronic system design, Concurrent Engineering: Research and Applications, 4(4):401-412.

Gebala, D. A., and Eppinger, S. D. (1991) Methods for Analyzing Design Procedures, Proceedings of the ASME Third International Conference on Design Theory and Methodology, pp. 227-233.

Genesereth, M. R. and Fikes, R. E. (1992) Knowledge Interchange Format, Version 3.0 Reference Manual, Technical Report Logic-92-1, Computer Science Deptartment, Stanford University.

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

Gero, J. S., Tham, K. W. and Lee, H. S. (1992) Behaviour: A link between function and structure in design, Brown, D. C., Yoshikawa, H. and Waldron, M. (Eds.), Intelligent Computer-Aided Design, North-Holland, Amsterdam, pp.193-225.

Gero, J. S. and Yan M. (1993) Discovering emergent shapes using a data-driven symbolic model, In Flemming, U. and Van Wyk, S. (EDs.), CAAD Features ’93, North Holland, Amsterdam, pp. 3-17.

Goldmann, S. (1996) Procura: A Project Management Model of Concurrent Planning and Design, Proceeding of WET ICE'96, Stanford, CA.

Gorti, S. R., Gupta, A., Kim, G. J., Sriram, R. D. and Wong, A. (1998) An object-oriented representation for product and design process, Computer-Aided Design, 30(7):489-501.

Greif, I. (1998) Computer-Supported Cooperative Work: A Book of Readings, Mogan Kaufman Press.

Gruber, T. R., Tenenbaum, J. M. and Weber, J. C. (1992) Toward a Knowledge Medium for Collaborative Product Development, Gero, J. S. (Eds.), Artificial Intelligence In Design, Pittsburgh, Kluwer Academic Publishers, pp. 413-432.

Grudin, J. (1994) Computer-Supported Cooperative Work: History and focus, IEEE Computer, 27(5):19-26.

Gu, N., Tang, M. X., Frazer, J. H. and Chai, X. L. (1999) Using Semantics to Describe Collaborative Design in CoDesign System, Proceedings of Fourth International Workshop on CSCW in Design, France, pp. 217-222.

Gu, P. and Chan, K. (1995) Product modeling using STEP, Computer-Aided Design, 27(3):163-179.

Gutwin, C. and Greenberg, S. (1996) Workspace Awareness for Groupware, ACM SIGCHI'96 Conference on Human Factors in Computing System, Companion Proceedings, pp. 208-209.

Hague, M. J. and Taleb, B. A. (1998) Tool for Management of Concurrent Conceptual Engineering Design, Concurrent Engineering: Research and Applications, 6(2):111-129.

Hague, M. J., Taleb, B. A. and Brandish, M. J. (1996) An adaptive machine learning system for computer supported conceptual engineering design, AI System Support for Conceptual Design, Proceedings of the 1995

Lancaster International Workshop on Engineering Design, London, UK, Springer.

Hales, L. and Lavery, M. (1991) Workflow Management Software: the Business opportunity, Ovum Ltd., London, UK.

Hao, Q., Shen, W., Zhang, Z., Park, S. W. and Lee, J. K. (2006) Agent-Based Collaborative Product Design Engineering: An Industrial Case Study, Computers in Industry, 57(1):26–38.

Hu, W. and Wang, S. (2005) Study on the architecture of product collaborative design system in mass customization base on J2EE, Proceedings of the Ninth International Conference on Computer Supported Cooperative Work in Design, 1:206-210.

Huang, G. Q. and Mak, K. L. (1999a) Design for Manufacture and Assembly on the Internet, Computers in Industry, 38(1):17-30.

Huang, G. Q. and Mak, K. L. (1999b) Web-based Collaborative Conceptual Design, Journal of Engineering Design, 10(2):183-194.

Huang, G. Q. and Mak, K. L. (2001) Issues in the development and implementation of web applications for product design and manufacture, International Journal of Computer Integrated Manufacturing, 14(1):125-135.

Hurst, K. S. (1999) Engineering design principles, J W Arrowsmith Ltd., London, UK.

ICSID (2004) Definition of Design, International Council of Societies of Industrial Design, http://www.icsid.org/.

ISO International Standards Organization, http://www.iso.org.

ISO (1991a) ISO 10303, STEP Part 21, Integrated resource: exchange of product model data.

ISO (1991b) ISO 10303, STEP Part 48, General resources: form features.

ISO (1991c) ISO 10303, STEP Part 11, The express language user manual.

ISO (1992a) ISO 10303, STEP Part 203, Application protocol: configuration controlled design.

ISO (1992b) ISO 10303, STEP Part 42, Integrated resource: geometric and topological representation.

ISO (1992c) ISO 10303, STEP Part 43, Integrated resource: representation structure.

ISO (1992d) ISO 10303, STEP Part 47, Integrated resource: shape variation tolerances.

ISO (1994a) ISO 10303, STEP Part 41, Integrated resource: fundamentals of product description and support.

ISO (1994b) ISO 10303, STEP Part 11, 21, 42 and 203.

JACK (2006) JACK Intelligent Agents, Agent Oriented Software Pty. Ltd., http://www.agent-software.com/shared/products/index.html

JADE (2006) Java Agent Development Environment, http://jade.tilab.com.

JATLite (2003) Java Agent Template Lite, http://java.stanford.edu.

Jones, J. C. (1984) A Method of Systematic Design, Cross, N. (Eds.), Developments in Design Methodology, Wiley, Chichester.

Keen, P. G. W. (1987) MIS research: current status, trends and needs, Buckingham, R. A., Hirschheim, R. A., Land, F. F. and Tully, C. J. (Eds.), Information systems education, Recommendations and implementation, Cambridge University Press, Cambridge, UK, Chapter 1, pp. 1-13.

Khoo, L. P., Chen, C. H. and Jiao, L. (2003) A Dynamic Fuzzy Decision Support Scheme for Concurrent Design Planning, Concurrent Engineering: Research and Applications, 11(4):279-288.

Kincaid, C., Dupont, P. and Kaye, A. (1985) Electronic calendars in the office: an assessment of user needs and current technology, ACM Transactions on Office Information Systems, 3(1):89-102.

Kunz, J. C., Tore R., Christiansen, T. R., et al. (1998) The Virtual Design Team: A Computational Simulation Model of Project Organizations, Communications of the Association for Computing Machinery (CACM), 41(11):84-91.

Lander, S. (1997) Issues in Multiagent Design Systems, IEEE Expert: Intelligent Systems and their Applications, 12(2):18-26.

Lee, K. C., Mansfield, W. H. J. and Sheth, A. P. (1993) A Framework for Controlling Cooperative Agents, IEEE Computer, July 1993, pp. 8-16.

Lesser, V. R., (1999) Cooperative Multiagent Systems: A Personal View of the State of the Art, IEEE Transactions on Knowledge and Data Engineering, 11(1):133-142.

Leviit, R. E., Christiansen, T. R., et al. (1994) The Virtual Design Team: A computational Simulation Model of Project Organizations, CIFE Working Paper.

Li, W. D., Lu, Y. Q., Zhou, H., et al. (2002) A Distributed Feature-based Environment for Collaborative Design, Proceedings of 6th World Multi-conference on Systemics, Cybernetics and Informatics, Orlando, Florida, US, Volume XII, pp. 69-74.

Li, W. D., Lu, W. F., Fuh, J. Y. H. and Wong, Y. S. (2005) Collaborative Computer-Aided Design – Research and Development Status, Computer-Aided Design, 37:931-940.

Li, Y. and Shen, W. (2002) CODA: A Collaboration Oriented Data Agent, Proceedings of the 7th International Conference on Computer Supported Cooperative Work in Design (CSCWD'2002), pp. 252-257.

Liu, H., Tang, M. X. and Frazer, J. H. (2001) Supporting learning in a shared design environment, Advances in Engineering Software, 32(4):285-293.

Liu, H., Tang, M. X. and Frazer, J. H. (2002) Supporting evolution in a multi-agent cooperative design environment, Advances in Engineering Software, 33:319-328.

Liu, H., Tang, M. X. and Frazer, J. H. (2004) Supporting creative design in a visual evolutionary computing environment, Advances in Engineering Software, 35(5):261-271.

Liu, Y. T. (1993) A connectionist approach to shape recognition and transformation, Flemming, U. and Van Wyk, S. (Eds.), CAAD Features '93, North Holland, Amsterdam, pp. 19-36.

Lyytinen, K. (1989) Computer Supported Cooperative Work: Issues and Challenges, Technical Report, Department of Computer Science, University of Jyvaskyla, Finland.

March, L. J. (1984) The Logic of Design, Cross, N. (Eds.), Developments in Design Methodology, Wiley, Chichester.

Mark d'Inverno, Kinny, D., Luck, M. and Wooldridge, M. (1997) A Formal Specification of dMARS, Lecture Notes in Computer Science, 1365:155-176.

Matchett, E. (1968) Control of Thought in Creative Work, The Chartered Mechanical Engineer, 15:163-166.

McCarthy, J. (1994) The state-of-the-art of CSCW: CSCW systems, cooperative work and organization, Journal of Information Technology, 9:73-83.

McGrath, M. E., Anthony, M. T. and Shapiro, A. R. (1992) Product Development: Success through Product and Cycle-time Excellence, The Electronic Business Series, Butterworth Heinemann, Stoneham, MA.

Meredith, D. D., Wong, K. W., et al. (1985) Design and Planning of Engineering Systems, Second edition, Prentice-Hall, N.J.

Mervyn, F., Kumar, A. S., Bok, S. H. and Nee, A. Y. C. (2004) Developing Distributed Applications for Integrated Product and Process Design, Computer-Aided Design, 36(8):679-689.

Mori, T., Ishii, K., Kondo, K. and Ohtomi, K. (1999) Task Planning for Product Development by Strategic Scheduling of Design Reviews, Proceedings of DETC'99, 1999 ASME Design Engineering Technical Conferences, pp. 115-126.

Ndumu, D. T., and Nwana, H. S. (1997) Research and Development Challenges for agent-based systems, IEE Proceedings On Software Engineering, 144(1):2-10.

NIST (1996) Initial Graphics Exchange Specifications (IGES), Version 5.3, More information is available on http://www.nist.gov/iges/.

Numata, J. (1996) Knowledge Amplification: An Information System for Engineering Management, Sony's Innovation in Management Series, Volume 17, Sony Corporation, Japan.

Nunamaker, J. F., Chen, M. and Purdin, T. D. M. (1991). Systems development in information systems research, Journal of Management Information Systems, 7(3):89-106.

Nwana, H. S. (1996) Software Agents: An Overview, Knowledge Engineering Review, 11(3):205-244.

Nwana, H. S., Ndumu, D., Lee, L. and Collis, J. (1999) ZEUS: A Tool-Kit for Building Distributed Multi-Agent Systems, Applied Artificial Intelligence, 13(1):129-186.

OOA (2005) Open Agent Architecture, http://www.ai.sri.com/~oaa/.

Oliveira, J. A. (2004) Coalition Based Approach for Shop Floor Agility - A Multi-Agent Approach, PhD thesis, Universidade Nova de Lisboa.

Olsen, G. R., Cutkosky, M., Tenenbaum, J. M. and Gruber, T. R. (1994) Collaborative Engineering based on Knowledge Sharing Agreements, ASME Database Symposium, Minneapolis, MN.

Pahang, G. D. F., Bae, S. and Wallace, D. (1998) A Web-based Collaborative Design, Modelling Environment, Proceedings of the IEEE Workshops on Enabling Technologies Infrastructure for Collaborative Enterprises (WET ICE'98), pp. 161-167.

Pahl, G. and Beitz, W. (1996) Engineering Design: A Systematic Approach, Second Edition, Springer-Verlag London Limited, UK.

Palen, L. (1999) Social, individual and technological issues for groupware calendar systems, Proceedings of ACM CHI'99 Conference, pp. 15-20.

Park, H. (1995) Modeling of collaborative design processes for agent-assisted product design, Ph.D. Dissertation, Stanford University.

Park, H. and Cutkosky, M. R. (1999) Framework for modeling dependencies in collaborative engineering processes, Research in Engineering Design - Theory, Applications and Concurrent Engineering, 11(2):84-102.

Park, H., Cutkosky, M. R., Conru, A., et al. (1994) An Agent-Based Approach to Concurrent Cable Harness Design, Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 8(1):45-62.

Parunak, H. V. D. (1998) What can agents do in industry, and why? An overview of industrially-oriented R&D at CEC, Cooperative information agents II: Learning, mobility and electronic commerce for information discovery on the Internet: Second International Workshop, CIA'98, Springer, Paris, France, pp. 1-18.

Parunak, H. V. D. (1999) Agents in Overalls: Experiences and Issues in the Development and Deployment of Industrial Agent-Based Systems, Cooperative Information Systems, 9(3):209-227.

Pels, H. J. (1996) Product and process data modelling: introduction, Computers in Industry, 31(3):191-194.

Perdikeas, M. K., Chatzipapadopoulos, F. G., Venieris, I. S. (1999) An Evaluation Study of Mobile Agent Technology: Standardization, Implementation and Evolution, Proceedings of the IEEE International Conference on Multimedia Computing and Systems, 2:287-291.

Petrie, C., Cutkosky, M. and Park, H. (1994) Design Space Navigation as a Collaborative Aide, Proceeding of the Third International Conference on AI in Design, Lausanne.

Petrie, C., Cutkosky, M., Webster, T., Conru, A. and Park, H. (1994) Next-Link: An Experiment in Coordination of Distributed Agents, Position paper, AID-94 Workshop on Conflict Resolution, Lausanne.

Popovic, V. (2003) General Strategic Knowledge Models Connections and Expertise Development in Product Design, Expertise in Design Symposium, UTS, Sydney, pp. 251-271.

Qiang, W. (2002) CSCW Bibliography, http://www.lehigh.edu/~qiw3/publication/CSCW Biblio.pdf.

Qureshi, S. M., Shah, J. J., Urban, S. D., et al. (1997) Integrated Model to Support Archival of Design Process History Databases, Proceedings of

the ASME 1997 Design Engineering Technical Conferences, Sacramento, California, CD-ROM index no. DETC97/DTM-3876.

Ramani, K., Agrawal, A. and Babu, M. (2003) CADDAC: Multi-Client Collaborative Shape Design System with Server-based Geometry Kernel, Journal of Computing and Information Science in Engineering, 3:170-173.

Reddy, R., Erman, L. D., Fennel, R. and Neely, R. (1976) The HEARSAY Speech Understanding System: An Example of the Recognition Process, IEEE Transactions on Computers, pp. 427-431.

Rosenman, M. A. and Gero, J. S. (1994) The what, the how, and the why in design, Applied Artificial Intelligence, 8(2):199-218.

Rosenman, M. A. and Gero, J. S. (1996) Modelling multiple views of design objects in a collaborative CAD environment, Computer-Aided Design, 20(3):193-205.

Rosenman, M. A. and Wang, F. (1999) CADOM: A Component Agent-based Design-Oriented Model for Collaborative Design, Research in Engineering Design, 11(4):193-205.

Rosenman, M. A. and Wang, F. (2001) A Component Agent Based Open CAD System for Collaborative Design, Automation in Construction, 10(4):383-397.

Roy, U. and Kodkani, S. S. (1999) Product Modelling within the Framework of the World Wide Web, IIE Transactions, 31(7):347-365.

Serrano, D. and Gossard, D. (1988) Constraint management, Gero, J. S. (Eds.) Artificial Intelligence in Engineering Design, Computational Mechanics Publications, New York.

Shen, W. and Barthes J. P. (1995a) An Object-Oriented Approach for Engineering Design Product Modelling, Knowledge Intensive CAD-1, Chapman & Hall, pp. 245-262.

Shen, W. and Barthes, J. P. (1995b) DIDE: A Multi-Agent Environment for Engineering Design, Proceedings of First International Conference on Multi-Agent Systems, Menlo Park, CA, AAAI Press, pp. 344-351.

Shen, W. and Barthes, J. P. (1996) An experimental environment for exchanging engineering design knowledge by cognitive agents, Knowledge intensive CAD-2, Chapman and Hall, pp. 19-38.

Shen, W., Norrie, D. H. and Barthes, J. P. (2000) Multi-agent systems for concurrent intelligent design and manufacturing, Taylor & Francis Inc., New York, NY.

Shen, W. and Wang, L. (2003) Web-Based and Agent-Based Approaches for Collaborative Product Design: an Overview, International Journal of Computer Applications in Technology, 16(2/3):103-112.

Shyamsundar, N. and Gadh, R. (2001) Internet-based collaborative product design with assembly features and virtual design spaces, Computer-Aided Design, 33(9):637-651.

Silaghi, G. C. (2005) Contributions to Conception, Design and Development of Collaborative Muliti-agent Systems, PhD thesis, Babes-Bolyai University Cluj-Napoca, Romania.

Simoff, S. and Maher, M. L. (1998) Designing with the activity/space ontology, Artificial Intelligence in Design 98 (AID'98), Kluwer Academic, Dordrecht, pp. 23-44.

Soley, R. M. and Stone, C. M. (1995) Object Management Architecture Guide, Third edition, John Wiley & Sons, New York.

Sriram, D., Logcher, R., Groleau, N., and Cherneff, J. (1992) DICE: An Object Oriented Programming Environment for Cooperative Engineering Design, AI in Engineering Design, Academic Press, 3:303-366.

Stephanopoulos, G., Johnston, J., Kriticos, T., et al. (1987) DESIGN-KIT: An Object-Oriented Environment for Process Engineering, Computer in Chemical Engineering, 11(6):655-674.

Steward, D. V. (1981a) The design structure system: A method for managing the design of complex systems, IEEE Transactions on Engineering Management, 28:71-74.

Steward, D. V. (1981b) Systems Analysis and Management: Structure, Strategy and Design, Petrocelli Books, New York.

Su, D., Li, J. and Ji, S. (2004) Online Collaborative Design within a Web-Enabled Environment, Lecture Notes in Computer Science, 3168:211-220.

Su, D., Li, J., Yu, X. and Zheng, Y. (2005) Collaborative Design and Manufacture Supported by Multiple Web/Internet Techniques, Lecture Notes in Computer Science, 3865:483-492.

Suh, N. P. (1990) The Principles of Design, Oxford University Press, New York.

Sycara, K., Paolucci, M., et al. (2003) The RETSINA MAS Infrastructure, Autonomous Agents and Multi-Agent Systems, 7(1/2):29-48.

Talukdar, S., Quadrel, R. and Christie, R., (1992) Multiagent Organizations for Real-time Operations, Proceedings of the IEEE, 80(5):765-778.

Tamine, O. and Dillmann, R. (2003) KaViDo - A Web-based System for Collaborative Research and Development Processes, Computers in Industry, 52(1):29-45.

Tang, M. X. (1997) A Knowledge-based Architecture for Intelligent Design Support, International Journal of Knowledge Engineering Review, 12(4):387-460.

Tang, M. X. and Frazer, J. H. (2001) A Representation of Context for Computer Supported Collaborative Design, Automation in Construction, 10(6):715-729.

Tang, M. X. and Frazer, J. H. (2004) Integration of Design and Technology: A Computational Approach, The Changing Face of Design Education: Proceedings of the 2nd International Engineering and Product Design Education Conference, pp. 1-7.

Toye, G., Cutkosky, M., Leifer, L., et al. (1993) SHARE: A methodology and environment for collaborative product development, Proceedings of Second Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, IEEE Computer Society Press, pp. 33-47.

VDI (1987) Design Handbook 2221, Systematic Approach to the Design of Technical Systems and Products (translated by K. Wallace), VDI-Verlag, Düsseldorf.

Vrba, P. (2003) JAVA-Based Agent Platform Evaluation, Holonic and Multi-agent Systems for Manufacturing and Control, Lecture Notes in Computer Science, 2744:47-58.

Wang, L. (1999) An approach to collaborative design and intelligent manufacturing, Proceedings of International Joint Conference of SCI'99 (Systemics, Cybernetics and Informatics) and ISAS'99 (Information Systems Analysis and Synthesis), 7:431-437.

Wang, L., Shen, W., Xie, H., Neelamkavil, J., and Pardasani, A. (2002) Collaborative Conceptual Design - State of the Art and Future Trends, Computer-Aided Design, 34(13):981-996.

Wang, Y. and Nnaji, B. O. (2004) UL-PML: Constraint-Enabled Distributed Design Data Model, International Journal of Production Research, 42(17):3743-3763.

Wiest, J. D. and Levy, F. K. (1977) A Management Guide to PERT/CPM, Prentice-Hall, Englewood Cliffs, N.J.

Wong, A. and Sriram, R. D. (1993a) SHARED: An Information Model for Cooperative Product Development, Research in Engineering Design, 5(1):21-39.

Wong, A. and Sriram, R. D. (1993b) Geometric modeling for cooperative product development, Proceedings on the Second ACM Symposium on Solid Modeling and Applications (SMA '93), New York, pp. 497-498.

Wong, K. S., Sellappan, P. and Nor, A. Y. (2005) A Framework for Collaborative Graphical Based Design Environments, Proceedings of The 5th International Workshop on Web Based Collaboration (WBC 2005), pp. 646-650.

Wooldridge, M. (2002) An introduction to multiagent systems, John Wiley & Sons Ltd.

Xiao, A., Choi, H. J., Kulkarni, R., et al. (2001) A Web-based Distributed Product Realization Environment, Proceedings of the 2001 ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Pittsburgh, PA, DETC2001/CIE-21766.

Yang, H. and Xue, D. (2003) Recent Research on Developing Web-based Manufacturing Systems: A Review, International Journal of Production Research, 41(15):3601-3629.

Ye, Y. M. and Elizabeth, C. (2003) Agent Supported Cooperative Work, Kluwer Academic Publishers.

Zha X. F. and Du, H. A. (2002) PDES/STEP-based model and system for concurrent integrated design and assembly planning, Computer-Aided Design, 34(14):1087-1110.

Zhao, G., Deng, J. and Shen, W. (2001) CLOVER: An Agent-based Approach to Systems Interoperability in Cooperative Design Systems, Computers in Industry, 45(3):261-276.