Negotiation Between Distributed Agents in a Concurrent Engineering System

by

Sundar K. Victor

A Thesis
Submitted to the Faculty
of the
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements for the
Degree of
Master of Science
in
Computer Science

by

August 5, 1993

APPROVED:

Prof. David C. Brown, Thesis Advisor

Prof. Robert E. Kinicki, Head of Department

To my parents, Mr. & Mrs. Victor Isaac, my parents-in-law, Mr. & Mrs. Arumai Singh, my wife, Shobi, and my daughter, Ruby, I gratefully dedicate this thesis. Thank you for always supporting me.

Acknowledgments

I would like to thank Professor David C. Brown for all his help. It was absolutely invaluable. I would also like to thank him for all those brain storming sessions we had together.

Thanks to Professor Becker for reading this mess, and for being so kind about it. Thanks to Dan Grecu for all the interesting technical discussions. Special thanks to the evaluation experts, Pete McCann and Professor John J. Bausch III of the manufacturing engineering department.

Also, thanks to Kathy Urbanowicz and Bertram Dunskus for suggesting corrections, and to Robert Douglas for his help with LaTeX .

Abstract

Current approaches to design are often serial and iterative in nature, leading to poor quality of design and reduced productivity. Complex artifacts are designed by groups of experts, each with his/her own area of expertise. Hence design can be modeled as a cooperative multi-agent problem-solving task, where different agents possess different expertise and evaluation criteria. New techniques for Concurrent Design, which emphasize parallel interaction among design experts involved, are needed. During this concurrent design process, disagreements may arise among the expert agents as the design is being produced. The process by which these differences are resolved to arrive at a common set of design decisions is called *Negotiation*. The main issues associated with the negotiation process are, whether negotiation should be centralized or distributed, the language of communication and the negotiation strategy. The goals of this thesis are to study the work done by various researchers in this field, to do a comparative analysis of their work and to design and implement an approach to handle negotiation between expert agents in an existing Concurrent Engineering Design System.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

This thesis is concerned with the use of negotiation in design expert systems. It consists of two parts. The first is to produce a comparative analysis of some of the existing work on negotiation, and to classify the various approaches. The second is to develop a Negotiation Demonstration System. The specific goals of this thesis are detailed in Section 1.2. The next two sections give an overview about Cooperative Problem Solving (CPS) and Concurrent Engineering (CE). The last section discusses the organization of the rest of the thesis.

1.2 Goals of the Thesis

There are two major goals for this thesis. The first goal is to explicitly identify the various factors that contribute to the negotiation process, and to do a comparative analysis of the existing work on an attribute-by-attribute basis, to classify those works. The second goal is to build a Negotiation System in which some of these negotiation techniques will be demonstrated. The I3D system [Victor et al, 1993], which does not use negotiation, will be extended by introducing conflicts among the various expert

agents, and resolving the conflicts using negotiation. The resulting system will be called the I3D+ System. The systems will be compared.

1.2.1 Comparative Analysis of Existing Work

Based on research, the various selected negotiation systems [Lander & Lesser 1992], [Sycara 1991], [Klein 1991], [Werkman 1992], [Kannapan & Marshek 1991], were compared and classified based on a set of attributes. This classification uniquely characterizes any given negotiation system. This classification of negotiation systems is "unique" and, as yet, has not been found in the literature.

1.2.2 Building a Negotiation System — I3D+

The Concurrent Engineering system, I3D [Victor et al, 1993], was extended by introducing conflicts among the various expert agents. The extended system is called the I3D+ System. When the agents in the I3D+ are scheduled to execute, conflicts could arise, and these conflicts are resolved using negotiation. The major claim to this work is that we are using a practical working concurrent engineering system in a real domain. In the existing, original concurrent engineering system, I3D, negotiation was avoided as all conflicts were avoided. In the I3D+ system, we deal with conflicts; we allow them to exist, and resolve them using negotiation.

1.3 Cooperative Problem Solving

Cooperative Problem Solving (CPS), [Lander & Lesser 1992], [Klein 1993b] a methodology in which problems are solved by groups of experts, is well studied in the domain of sociology, organizational science, public policy and international relations. A central aspect of cooperative problem solving by groups is the avoidance, detection and resolution of conflicts among the participants. This is of great theoretical interest in such research areas as Distributed Artificial Intelligence (DAI) [Davis & Smith 1983]. It is also of considerable practical importance because of the key role conflict management plays in cooperative problem solving, e.g., concurrent engineering [Brown & Douglas 1993b]. Work on conflict management has occurred in

a variety of settings including concurrent engineering [Subramanian et al 1990], multi-agent planning and design [Sycara 1990c], Group Decision Support System (GDSS), Computer Supported Cooperative Work (CSCW) and Software Engineering.

The CPS domain thus includes theoretical groundwork, empirical studies and implemented conflict management systems for human and computational agents. There are many areas of research open in this CPS domain, and some of the general issues that have to be addressed are:

• What lessons do empirical studies of conflict management have to offer for the development of computational models?

• What are the current theoretical underpinnings of conflict management, and how can they be applied to practical problems?

• How can computers support group conflict management with both human and computational participants? What are the benefits and challenges of the different approaches?

• What aspects of conflict management are generic and what are domain-specific? Can the same techniques work with human and computational participants?

• How do computational models of conflict management fare in real-world social and organizational settings?

1.4 Concurrent Engineering

Concurrent Engineering (CE) is a new design methodology, which enhances productivity and leads to better overall designs. I will give a brief introduction to CE, although a more sizable overview is given in [Brown & Douglas 1993b] and [Bedworth et al 1991], while a much more concise summary is given in [Altamuro 1991].

In the development of a product, there are many "downstream" aspects to be considered. These include final cost, manufacturability, inspectability, reliability, and durability. These aspects represent different phases in the product's life-cycle. In traditional design methodologies, the product is evaluated after each phase is complete. However, the downstream issues are affected, perhaps negatively, by decisions made during the design phase. Consequently, these aspects should be taken into account during the design phase.

In the CE scheme, these aspects positively affect the design decisions during the design phase. A team, composed of experts on each aspect, is brought together to participate in the design. These people have information about how downstream issues are affected by design decisions. Having information about downstream issues at design time has several advantages. First, having all of this information minimizes the possibility of needing to redesign some or all of the product. Eliminating redesigns cuts product development time and cost. Next, decisions which take advantage of particular features of an aspect can be made, such as choosing a set of parts in the design for which the manufacturing equipment is already tooled. Stoll [Stoll 1986] gives an overview of these issues as they apply to the manufacturing aspect of product development. The practice of considering manufacturing needs at design time is known as Design For Manufacturing (DFM). CE attempts to extend the DFM principle to other aspects of the product's life-cycle.

Another advantage of CE is that, while knowledge is being built up about the design of the product, additional knowledge is being acquired about the other aspects of its life-cycle. As the design progresses, the manufacturing expert will learn more about how to manufacture the product, and the packaging expert will know more about how to package it, etc. This accumulation of knowledge helps to speed the product through the development process and get it to the customer more quickly, i.e., time-to-market is reduced.

1.4.1 I3D: A Concurrent Engineering System

The I3D system [Victor et al, 1993], shown in Figure 1.1, interacts with a designer sitting at a workstation. As the designer moves through requirements specification, conceptual design and detailed design of a part to be made from powder ceramic material, the system graphically displays the state of the design on the screen. It makes appropriate assumptions about design decisions not yet made in order to be able to continuously display the component during both the conceptual and detailed stages of design.

As requirements are given and design decisions are made, the system provides feedback about the design from several different points of view. Intelligent agents,

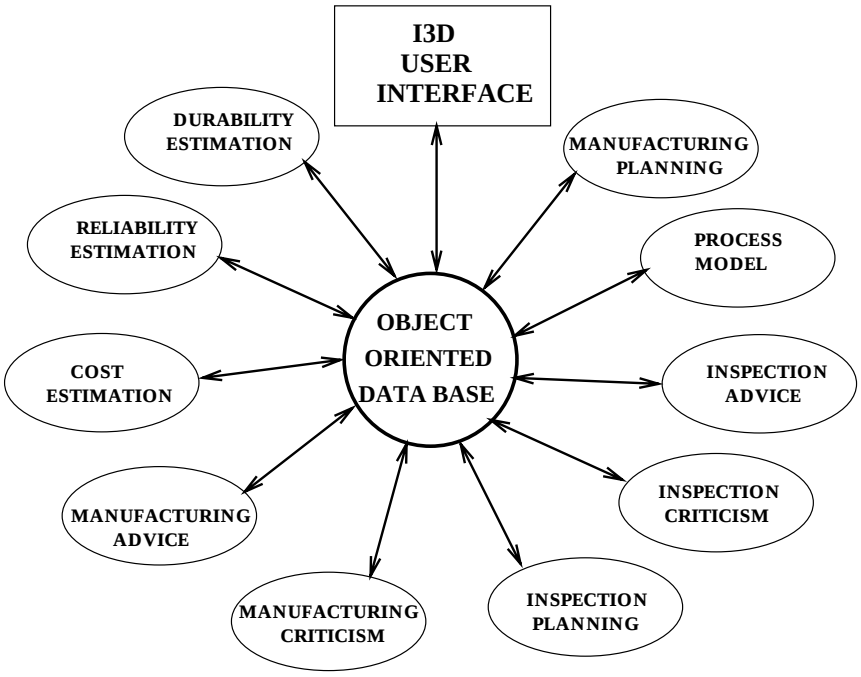

Figure 1.1: The I3D System Architecture

expert systems, display these comments on the screen. Each agent is given a chance to respond. Comments might include estimation of cost, advice about which ceramic powder to use, and information about the inspection process required for the part.

The system is intended to be extensible, so every effort was made to allow different agents to be added easily. To test this, we added intelligent agents, as well as adding an agent which performed a simulation to determine the effects of compaction and sintering on part size, density and cracking. Additions were found to be easy.

The system was mainly intended to be an investigation of what knowledge, what types of agents, and what type of control strategy would be required for an interactive design system of this kind for powder ceramics. It was not intended as a "complete" concurrent engineering system. Consequently we concentrated on many aspects of the design process, but for a limited class of parts.

A unique feature of the I3D system is that it can provide considerable feedback about a variety of "downstream" aspects during the conceptual design phase. For example, one agent performs cost estimation during the conceptual design phase, while

another estimates cost during the detailed design phase. Although the cost during the conceptual phase is approximate, it does provide useful information, allowing alternative designs to be compared.

1.5 Negotiation

Design is a problem-solving process based on multiple and diverse sources of expertise. Currently, it is often accomplished by a group of experts sequentially asserting and critiquing partial design decisions from their local perspectives. This process is very time consuming, often leads to very poor quality in the designed artifact and is not very economical.

Better design techniques are needed to achieve a high quality of design and to decrease the cost associated with it. To achieve this we need to do *concurrent design* rather than sequential interaction among the multiple sources of expertise. By doing concurrent design we can reduce the time it takes to market a product, cut cost due to the overhead associated with moving from one phase to another, and improve quality by bringing in various types of expertise simultaneously.

In a concurrent design environment, where multiple agents with their own sources of expertise interact, conflicts could arise among these expert agents. For example, in the I3D concurrent engineering system, the material agent could select a particular material (e.g., Silicon Carbide) using the functional requirements given by the user, but, from the cost agent's perspective, there could be objections to this choice of material, because it will increase the overall cost of the product. The important issue is how to resolve these conflicts among these expert agents. The process of resolving conflicts among these expert agents by exchanging information is called *Negotiation*.

1.5.1 Attributes

There are three main sets of attributes associated with the negotiation process. The first is concerned with the negotiation mechanism itself. The attributes include: what kind of a computational model is used for the negotiation process; what the various negotiation strategies are; whether these strategies are pre-determined or

dynamically adapted to the situation.

The second set of attributes involves the communication aspects of the negotiation process. Is there a communication protocol? Is there a separate communication language for the negotiation process? What is the communication medium being used?

The third set of attributes is concerned with the expert agents involved in the negotiation process. Here issues include: the types of agents; agent grainsize; what kind of domain knowledge the agents contain; agent privacy, such as how much information each agent knows about its counterparts (i.e., ignorant, partial or explicit).

1.6 The Thesis

The remaining chapters are organized as follows. Chapter 2 discusses other researchers' work in the area of negotiation and conflict resolution. Chapters 3, 4, and 5 compare existing work on Negotiation. Chapter 6 highlights the choice of the domain, the reasons behind the choice, the I3D System and the extensions made to this system. Chapter 7 details the implementational issues associated with the I3D+ system. The last chapter lists results, conclusions and suggestions for future work in this area. The appendices at the end include some traces from the I3D+ system.

Chapter 2

Literature Review

2.1 Introduction

Surveying the available literature, we find numerous approaches and tools developed by the various researchers in this field. From this survey, we have mostly concentrated on the research done by Dr. Katia P. Sycara at Carnegie Mellon University, Dr. Susan Lander and Professor Victor R. Lesser at the University of Massachusetts, Dr. Mark Klein at Boeing Computer Services, Dr. K.J. Werkman at IBM, and Dr. Srikanth M. Kannapan at Xerox Design Research Institute, Cornell University. The reason for concentrating on these research work is because, they have been referenced the most.

2.2 Sycara's Research

2.2.1 Overview

In Section 2.2.2 we briefly present Sycara's design process. In Section 2.2.3 we present her negotiation model, which incorporates communication of design rationale and criticisms of design decisions made by the expert agents. In her model, design modifications are made based on constraint relaxation and comparison of utilities.

In section 2.2.4 we present the negotiation process in which the agents iteratively exchange proposals and proposal justifications until an agreement is reached.

According to Sycara, negotiation consists of three main tasks: generation of a proposal, generation of a counterproposal based on feedback from dissenting parties, and communication of justifications and supporting evidence. An initial compromise is generated and presented to each expert. Each agent evaluates the proposal from its own point of view and registers its reactions (evaluations, objections and suggestions). The process terminates when all concerned agents accept a proposed design. Sections 2.2.5, 2.2.6, 2.2.7, 2.2.8 explain how the agents knowledge/expertise is represented, the interaction between agents, the negotiation between agents and the negotiation protocol, respectively.

2.2.2 Design

According to Sycara [Sycara 1991], design systems should have the ability to:

- represent designs in ways that facilitate solution construction;
- represent design records so as to facilitate explanation and design reuse;
- represent designs from multiple viewpoints;
- organize large reusable design knowledge bases.

Design records play an important role in design. A design record includes descriptions of the problem specification (design goals and constraints), the solution of the design problem, and the trace of decisions that shows why the solution satisfies the problem specifications. The design record should be organized in such a way that it will aid tasks such as explanation, redesign and design by analogy.

Design cases are used to store memories of the design processes along with relevant decisions and their justifications, and also to index these designs in terms of salient features. Failures and failure reasons are also stored so that they can be used to predict and avoid future failures. If features in the past situation that resulted in failed solutions are present in the current situation, then the failed solution should not be tried. If repairs are also stored along with the associated failure, the repair can be applied if the same failure occurs.

Because of the complexity of the design activity, it could be viewed as a problem-

solving process among cooperating expert agents. The final design is a cooperative effort, because each expert has insufficient local knowledge to solve the problem independently. However, conflicts arise due to the different knowledge and evaluation criteria each agent brings to the design process. It is difficult to resolve these conflicts due to the fact that experts do not have the same mental model of the design, and also they may not speak the same language. These differences leads to misunderstandings and long iterations of explanation between agents.

Hence Sycara's work focuses on modeling the process of reconciling design decisions and design proposals that arise from different agents perspectives during the design process in order to form an acceptable final design. The final successful design can be viewed as a compromise that incorporates tradeoffs such as cost, ease of manufacturing and assembly, reliability and maintainability.

Sycara's argument is that understanding the negotiation process in design will enable 1) the development of intelligent and efficient design support systems to aid human designers, and 2) the development of systems that can reason from design specification towards candidate solution structures. According to her, negotiation between agents involves finding a compromise solution for multiple conflicting goals, and is not amenable to traditional AI planning techniques. The negotiation process is dynamic; consequently formulating it as a search problem is inadequate since there are no well-defined goal or search operators. Also, hierarchical decomposition of the problem into smaller subproblems may not be suitable, since a compromise solution may be a package whose parts are strongly interconnected and interacting [Sycara 1991].

2.2.3 Negotiation Model

Negotiation occurs recursively at all levels and stages of design from conceptual design through detailed design. Negotiation enters the design process when different specialists have made conflicting recommendations about some attribute value or when an attribute value proposed by one expert makes it infeasible for another expert to offer a consistent set of values for other attributes. Also, negotiation enters the design process when one specialist has negative criticism about the decision made by

another specialist.

Negotiation is a process in which the parties iteratively exchange proposals and proposal justifications until an agreement is reached. The final design is represented as a compromise solution among the different specialists. Some of the characteristics of the negotiation process are:

• **The process is iterative.** Negotiation involving multiple agents with multiple conflicting goals/issues/assertions is usually a lengthy and iterative process. The parties start by having conflicting goals/issues/assertions and whose distance has to be narrowed gradually to zero. Hence, a negotiation model must be *iterative* rather than one shot.

• **The process requires feedback among agents.** After each round of proposals the agents give feedback to each other about which parts of a proposal they agree or disagree on. Hence, a negotiation model needs to be able to receive and evaluate *feedback* about a proposition.

• **The process must be able to propose modifications.** In order to arrive at an agreement, design proposals must be modified. Hence a negotiation model must have the ability to propose suitable *modifications*.

• **The process must be able to evaluate proposals.** Since final agreement is reached through narrowing their difference in the proposals of the parties, a negotiation model must have a way of *predicting/evaluating* whether each new proposal indeed narrows these differences.

• **The process must be able to generate justifications and arguments.** Reaching an agreement through negotiation entails that each of the parties must modify partially or totally some of their goals and proposals. A good reason for such modifications is justifications and arguments in support of or against proposed modifications. Hence, a negotiation model needs to have a component that generates *justifications and arguments*.

2.2.4 Negotiation Process

The Negotiation Process consists of three main tasks: generation of a proposal, generation of a counterproposal based on feedback from dissenting parties, and com-

munication of justifications and supporting evidence. An initial proposal is generated and presented to each agent, which is then evaluated by them from their point of view and register their reactions (evaluations, objections and suggestions). The process terminates when all concerned agents accept a proposed design.

Each agent during negotiation engages in the following activities:

• **Recommends design decisions**, i.e., designs that express potentially acceptable compromises and tradeoffs of the parties. Sycara is investigating the generation of design recommendations using a combination of case-based reasoning, utilities and constraint propagation techniques.

• **Justifies recommendations**. Often the agents with their own knowledge, cannot recognize why a proposed design may be the best under the given circumstances. In order for a design proposal to become intelligible and have an increased chance of being accepted, justifications must also be communicated.

• **Explore feasible alternatives** so as to optimize the proposed compromise. A memory of the past designs from the design cases provide a rich repository of such alternatives.

• **Modifies a rejected compromise** to make it more acceptable to the rejecting party without making it unacceptable to the party that had previously accepted it. This is done using previous cases and modification rules.

The input to the negotiation process is 1) the set of conflicting goals and violated constraints of the various design agents and 2) the context of the design. The output is either a single set of consistent design decisions that have been agreed upon by the agents, or an indication of failure if the negotiating parties did not reach agreement within a particular number of proposals. The final output is reached through an iterative process of proposal generation, justification and critiquing of the proposal, and repair and improvement of a rejected proposal.

In Figure 2.1, an agent's actions during the negotiation process is shown. The rectangles represent the negotiation planning process: plan generation, plan evaluation, plan presentation, argumentation (justifications and arguments), plan modification and memory updates (success or failure). The negotiation process starts with the "Generate Plan" rectangle. Each negotiation task (proposal generation, argumentation, proposal modification) uses a subset of the problem-solving process, namely plan

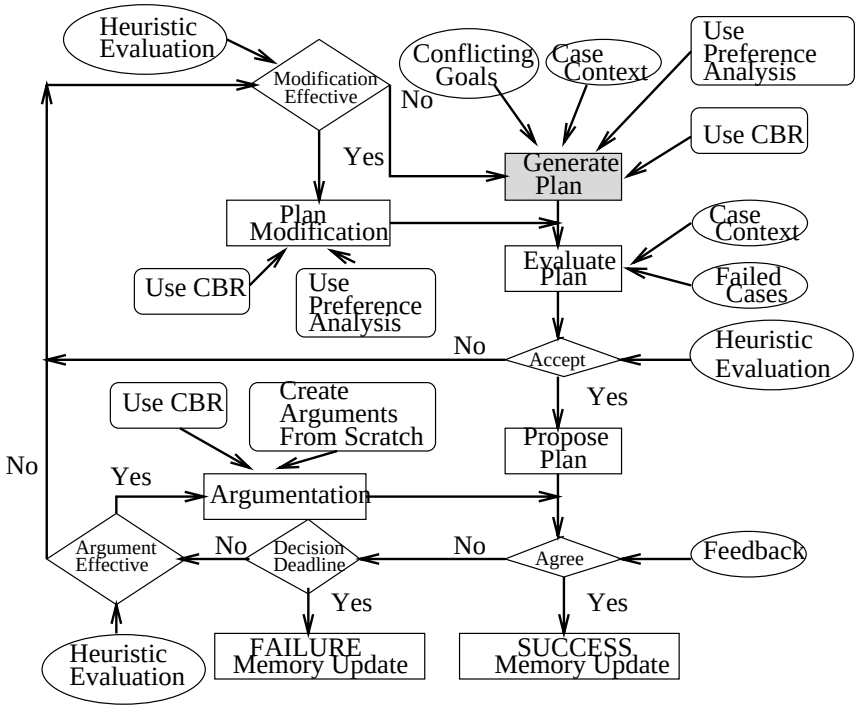

Figure 2.1: Sycara's Negotiation Process

generation, plan evaluation, plan presentation (to the agents) and plan modification (if needed). The word "plan" is used to denote a set of design decisions and applies to a different entity for each task. In the case of proposal generation, "plan" refers to a design proposal; for argumentation, "plan" refers to a set of justifications; for proposal modification, "plan" refers to a set of design decision modifications. The rectangles with rounded corners denote the techniques, CBR or multi-attribute utilities (Preference Analysis), used in each process. The ellipses denote input to the process.

Negotiation is performed through integration of case-based reasoning (CBR), multi-attribute utilities and constraint relaxation. These methods are employed in all the negotiation tasks (proposal generation, argumentation, proposal modification). The integration of heuristic and analytic methods makes the system both robust and flexible. The problem solver does not break down when heuristic methods fail; instead it has the flexibility to use whatever method is more natural to the particular situation.

2.2.5 Representation of Agents (Goals and Knowledge)

Exchange of proposals and justifications lies at the heart of negotiation. This is the process that is used to cohere the decisions of agents and guide the process towards solution convergence. Sycara claims [Sycara 1991] that in order to negotiate effectively, agents need the ability to:

1. represent and maintain models of the knowledge and goals/beliefs of other agents

2. reason about other agent's goals/beliefs

3. influence other agent's beliefs and intentions through the exchange of missing information, justifications and arguments.

During the process of communication of justifications and arguments, an agent reasons about another agent using its own model of that other agent, finds as many ways as the model will allow to affect the other agent's outcomes, and uses them selectively to influence the other agent.

Now we will look at the representational mechanisms that are used to 1) struc-

ture the knowledge that each agent has of its own goals, constraints and utilities and 2) model the goal and utilities of other agents. The models of the other agents that each agent maintains and refines are used to generate inferences about potential acceptability of proposals, arguments that are presented to other agents. To perform generation of arguments and justifications in the design domain an agent needs knowledge of previous designs, and knowledge of an agent's belief and goal structure.

Belief and Preference Structure

The belief structure of an agent consists of a collection of goals/beliefs, a sign (+ or -) that indicates goal importance, a goal importance rating, amount and feasibility (of achieving that goal) as well as relationships among goals. The word "belief" is used to indicate what is commonly thought of as beliefs, e.g., safety is good. The word "goal" indicates things such as, increased marketability, reduced operational costs, etc. The agent's expertise is represented as part of this belief/preference/goal structure as a directed acyclic graph (DAG), where each node represents an agent's goal. The edge connecting two goals represents the relationship between goals in terms of how one affects, either positively or negatively, the achievement of the other.

Sycara [Sycara 1991] considers an example, the process of designing a turbine blade. The agents involved are aerodynamics, structural engineering, manufacturing and marketing. In Figure 2.2, a partial view of the belief structure of the structural and aerodynamic agents are shown. In the figure we have only shown the edges connecting particular nodes, and the appropriate sign for simplicity. By traversing the goal graph we can answer, *which goals are supported by a set of design decisions and which design decisions are justified by a set of goals.* A path from node X to node Y in a goal graph constitutes a causal/justification chain that provides an explanation of the change in Y in terms of the change in X, assuming no other change has occurred in the rest of the graph. For example, from the point of view of structural engineering, decreasing the length of the blade, Blade-Length(-), decreases tensile stress, Tensile-Stress(-), which results in increased structural soundness, Structural-Soundness(+). Increase in structural soundness, increases reliability, resulting in increased safety and contributing to increased marketability of the blade.

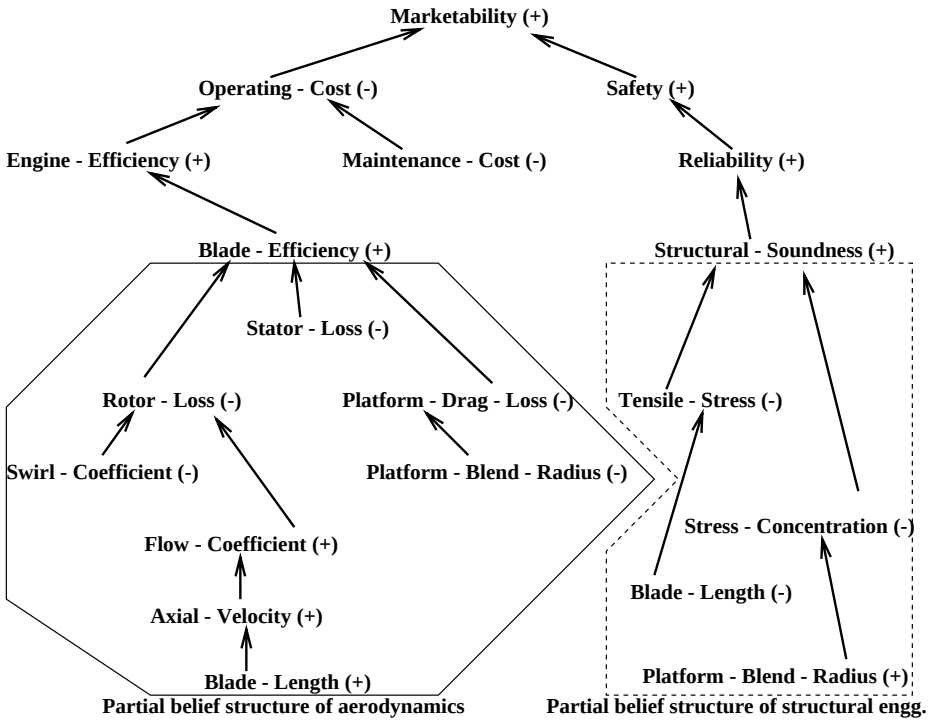

Figure 2.2: Sycara's Partial view of private expertise and shared communication vocabulary

The representation also includes an estimate of an agents utilities for each attribute in the goal graph, in addition to its beliefs. Utilities express the *preference structure* of an agent. Utilities also express the tradeoff knowledge among various attribute values associated with an alternative design. The utilities of individual attributes are combined to give an overall utility of an alternative design. Being able to compare different alternatives enables a reasoner to choose the alternative that affords the maximum payoff [Sycara 1988a].

2.2.6 Interaction between Agents

A central task in negotiation is communication, which is needed to exchange information between agents. Local computations are interleaved with agent communication. Output from these computations, such as, suggestions, evaluations of design proposals, are communicated to other agents. Agents could use this feedback from other agents as input to their computations. When considering communication, an interesting issue is the vocabulary used by the agents for communication. Experts neither share other expert's vocabulary nor understand other expert's problem-solving process. Hence, we need some means of communication between them. Sycara claims [Sycara 1991] that experts use an *intermediate* shared vocabulary to communicate with each other.

Even though each agent's expertise is private, the intermediate vocabulary is the medium for making public relevant portions of results from the agent's expertise. In the example shown in Figure 2.2, of designing a turbine blade, although the marketing expert might not understand the concepts of Axial-Velocity or Swirl-Coefficient, he understands the concepts of Blade-Efficiency and Structural-Soundness and how they relate to its own high level goal of marketability. Blade-Efficiency and Structural-Soundness are examples of terms in the intermediate vocabulary. In Figure 2.2, the bounded boxes indicate the private expertise of the aerodynamics and structural engineering experts, whereas the unbounded portion indicates terms to indicate goals and issues in the shared, public vocabulary.

2.2.7 Negotiation between Agents

The messages the negotiating agents exchange contain information such as, proposed design, justifications of design decisions, agreement/disagreement with a proposal, requests for additional information, reasons for disagreement or utilities/preferences of the agents associated with disagreed upon issues. Since different agents evaluate designs using different evaluation criteria, the information communicated by an agent to others cannot be simply its decisions. It needs to communicate justifications of its own design decisions and proposed design changes. If challenged, an agent must communicate justifications in support of its decisions.

Proposals and supporting justifications are used by an agent, the *persuader* as a means to dynamically change the utilities associated with the various decisions and outcomes of another agent, the *persuadee*, so as to increase the willingness of the persuadee to accept a proposal [Sycara 1991]. Now, by observing the reactions to the proposal and justifications, the persuader can update and correct its model of the persuadee, thus refining its planning and argumentation knowledge. In her work, proposal generation, counterproposals and justifications are based on the integration of *goal graph search*, *multi-attribute utilities* and *case memory*.

During negotiation an agent's belief structure is updated based on its reaction to presented new information and proposals. In this way, an agent's model is refined and corrected dynamically. This capability is important because it is not possible for an agent to have a correct and detailed belief model of another agent, and also beliefs are not static and change with external circumstances and agent's experiences. If an agent could manipulate another agent's utilities, he would be able to affect predictably the outcome of the second agent. Convincing an agent to change his evaluation and increase his willingness to accept a design decision is modeled as producing a justification which increase the payoff for that proposition.

The elements of a design decision are a subset of the information that appears in the agents belief structure. Hence the task of a persuader can be viewed as finding the most effective justification/argument that will increase a persuadee's payoff. Since a persuadee's payoff can be approximated by a linear combination of his utilities, the payoff can be increased either by changing the importance (coefficient) the persuadee

attaches to an issue, or changing the utility value of an issue.

The *argumentation goals* of a persuader express what in the beliefs and outcomes of a persuadee he wants to influence. To accomplish the argumentation goals, *argumentation strategies* are used. Based on a utility view of argumentation, there are two argumentation goals that could be used in the model: Change the importance of a persuadee's goal/issue; Change the persuadee's perception of an issue value [Sycara 1991]. If a persuadee disagrees with a proposed argument, the reasons for the disagreement are analyzed for new information that could alter subsequent argumentation, such as new information about the persuadee's concerns. If the analysis, done by the persuadee, reveals that the persuader had some incorrect notions regarding the beliefs and preferences of the persuadee, the appropriate updates are made to the persuadee's model. In addition, updates to the persuaders argumentation goals and strategies may be needed.

2.2.8 The Negotiation Protocol

The communication protocol presented here is still under investigation by Sycara. Here the protocol is simplified and presented for two agents. Agent 1 (persuader) initiates an initial design and Agent 2 (persuadee), evaluates the design and possibly generates a counterproposal. The eight steps followed in this protocol exchange are shown below:

1. Agent 1 communicates to agent 2 a design proposal, arguments and justification in support of the proposal.

2. Agent 2 uses the arguments and justifications communicated by agent 1 to possibly modify its goal graph.

3. Agent 2 evaluates the proposal from its point of view (constraints and utilities).

4. If the proposal satisfies agent 2's local constraints and gives it payoff above a threshold, it communicates AGREE to agent 1.

5. If not, agent 2 generates a counterproposal using CBR, constraint relaxation, and utilities/preferences.

6. Agent 2 now evaluates the counterproposal. If the counterproposal gives agent 2 payoff above the threshold, agent 2 communicates to agent 1, the portions of the

proposal that have been modified, the reasons for modifying the previous proposals, the counterproposal and its payoff, and the arguments and justifications in favor of the counterproposal.

7. If the counterproposal does not give agent 2 payoff above the threshold, agent 2 goes to step 5.

8. If agent 2 has exhausted all counterproposals it can generate through the methods of step 5, it communicates failure to agent 1, who now has to generate a modification or look for alternative proposals in its goal graph.

2.2.9 Summary

In Section 2.2, we examined Sycara's contribution towards understanding the multi-agent negotiation process in design problems. She has proposed a negotiation model that captures the dynamic interactions of the cooperating agents during negotiation. She clearly explained the negotiation process as shown in Figure 2.1. She also explained how an agent's knowledge/expertise as well as their model of other agents, are represented. She also explained how the agents interact during the negotiation process and what kind of a protocol they follow.

2.3 Klein's Research

2.3.1 Overview

In Section 2.3.2, we briefly talk about Klein's views on cooperative design and the problems associated with it. In Section 2.3.3 we describe his research on Conflict Resolution. In Section 2.3.4 we address the issue of how to organize conflict resolution expertise. We also present Klein's Taxonomy of conflict classes, used to represent the conflict resolution expertise. In section 2.3.5, we explain Klein's computational model for resolving conflicts which is based on studies of human cooperative design. His model is also based on insights that general conflict resolution expertise exists separately from domain-level design expertise, and that this expertise can be instantiated in the context of particular conflicts into specific advice for resolving those conflicts. In section 2.3.6 we present Klein's evaluation of his computational model,

for an implemented cooperative design system to design Local Area Networks (LAN).

2.3.2 Cooperative Design

Design has become an increasingly cooperative activity carried out by multiple expert agents with diverse kinds of expertise. To design a car, experts are needed to design function, ease of manufacturability, safety, packaging and marketing. A critical component of cooperative problem solving (CPS) is how conflicts among the different expert agents can be resolved. Different agents have different perspectives concerning what kind of a design is best. One agent may specify a particular shape in order to optimize strength, while another agent could recommend a different shape in order to ease the production process. Thus conflicts occur, when two experts give incompatible specifications for a given design component, or one expert has a negative critique of another experts specifications.

Klein [Klein 1991] points out that while conflict-free cooperation among multiple expert agents has been relatively well-studied, cooperation where conflict can occur is less well-understood. His computational model, gives first-class status to conflict resolution expertise. This expertise is instantiated in the context of a particular conflict, via interaction with domain-specific expertise, to produce suggestions for resolving that conflict. He also addresses his solution in the domain of cooperative situations, in which the agents are united by the superordinate goal of achieving a globally optimal solution which often requires sacrificing personal benefit in the interest of increased global benefit. His work mainly deals with domain level conflicts, concerning recommendations about the actual form of the design rather than control level conflicts concerning recommendations about the direction of the design process.

2.3.3 Conflict Resolution

The theory underlying his work is based on insights derived from studies of human cooperative design in two different domains: Solar Home Design and Local Area Network (LAN) Design. His fundamental belief is that conflict resolution expertise should be treated as a distinct, explicit, separate (first-class status) category of problem solving expertise for it to be used effectively. On this basis

he has grouped the available literature on conflict-resolution into three categories, *Development-Time Conflict Resolution, Knowledge-Poor Run-Time Conflict Resolution and General Conflict Resolution* . In the first category, *Development-Time Conflict Resolution,* all potential conflicts are compiled out by virtue of exhaustive discussion when they are developed. In addition, this approach makes the unrealistic assumption that human design agents in a cooperative design system will never make assertions that conflict with those made by other agents. In the second category, *Knowledge-Poor Run-Time Conflict Resolution,* conflicts are allowed to be asserted by the design agents as the systems run, and then resolved by some conflict resolution approach using either back-tracking, numerically-weighted constraint relaxation or pieces of specific conflict resolution advice [Brown 1985]. In the third category, *General Conflict Resolution,* conflict resolution expertise is given first-class status. Here conflict resolution expertise is made explicit and used to support cooperative problem solving (CPS) as explained in Section 1.3.

To drive home the point, he draws analogy with earlier knowledge-based systems in which domain expertise and control expertise are combined, via a "cross-product" like process to produce a more complicated combined body of expertise where neither domain expertise or control expertise are available in their original form. Since such knowledge base systems do not make the domain/control expertise explicit, they are difficult for domain experts to express, understand and modify. Current Knowledge-based systems make it possible to express domain/control expertise explicitly and succinctly. The advantage of having explicit control expertise is that it aids in deciding what kind of control scheme is appropriate for a given situation. This separation has resulted in increased flexibility and generality of knowledge-based systems.

He makes the same kind of argument for the separation of domain/conflict-resolution (CR) expertise. Currently all knowledge-based systems mix domain expertise with conflict-resolution expertise into a single knowledge base, resulting in the same kind of problem as described above. He says that giving Conflict Resolution expertise distinct *first-class status* allows us to capture succinctly general conflict resolution principles, and allows bodies of domain expertise to be represented in a pure form without having to anticipate potential conflicts with each other.

Now that he decided to represent conflict resolution expertise separately, what

is the nature of this expertise? Because of the common nature of conflicts, people have accumulated through experience a large collection of strategies, both specific and general for resolving conflicts in ways that are as satisfying as possible for all the parties involved. From the examples in [Klein 1991], conflict resolution instances can be thought of as instances of certain conflict resolution strategies. The conflict resolution strategies could be viewed as consisting of preconditions that match a given class of conflicts; and advice for how to resolve conflicts in that class. A given strategy can be instantiated for a wide variety of conflicts.

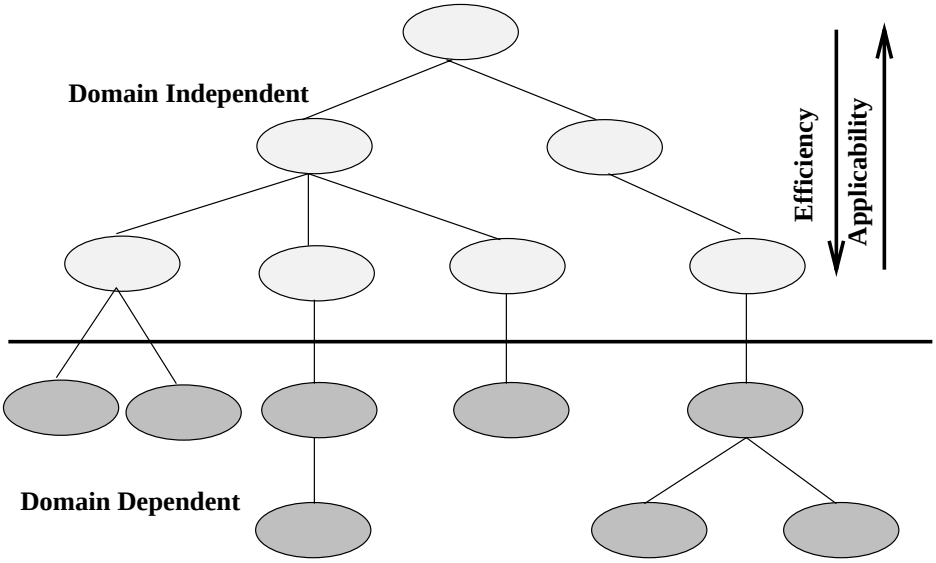

Figure 2.3: Klein's Conflict Resolution Hierarchy

2.3.4 Organizing Conflict Resolution Expertise

Conflict Resolution strategies can be organized using the notion of a conflict taxonomy. In particular, he believes that the different kinds of conflicts can be arranged into an abstraction hierarchy of conflict classes, and that we can associate applicable conflict resolution strategies with each conflict class. More general classes of conflict appear near the top of this *conflict hierarchy*, and more specific classes near the bottom, as shown in Figure 2.3. The more abstract classes represent domain-independent classes and associated strategies, while the more specific classes apply

only to particular domains. Conflict resolution strategies associated with more specific conflict classes will have a narrower range of applicability but usually greater efficiency than the more general strategies associated with abstract conflict classes. Thus CR expertise exhibits an applicability versus efficiency trade-off [Klein 1990].

An important advantage of this hierarchical arrangement is that it allows us to determine the range of conflict resolution strategies applicable to a given conflict. When a conflict occurs, we can find the most specific conflict class that subsumes that conflict, and try the CR strategies associated with that class. If none of these strategies are successful, we can climb the hierarchy to the next conflict class and try the more general, less efficient strategies associated with that class. A related advantage is that a conflict hierarchy can be useful even if one has not encoded specific strategies for every conflict that can occur. If there is no specific conflict class for that particular conflict, one can find the first more abstract conflict class that covers that kind of conflict and apply it.

CR expertise is heuristic in nature since it deals with the interaction of internally consistent but mutually inconsistent domain theories in different design agents. When we choose a particular strategy for a conflict, we are in effect making the hypothesis that the conflict is one that can be addressed by the given piece of advice, and the strategy must be able to respond appropriately if the advice fails. In addition to a set of CR strategies, CR expertise also includes control knowledge for determining which of a potentially large set of applicable CR strategies should be tried first for a particular conflict. Like CR strategies themselves, this kind of expertise has proven to be specific to most classes of problem domains [Klein 1991].

2.3.5 The Computational Model

The agents in Klein's cooperative design system can be viewed as being made up of a design component that can update and critique design decisions, and a conflict resolution component that resolves the design agent conflicts. This separation is due to the fact, that CR expertise is functionally distinct from domain-level design expertise. The conflict resolution component of all design agents are identical and replicated among the design agents to make the agents autonomous. The agent (its

CR component) that first detects a given conflict is given the lead in the conflict resolution process.

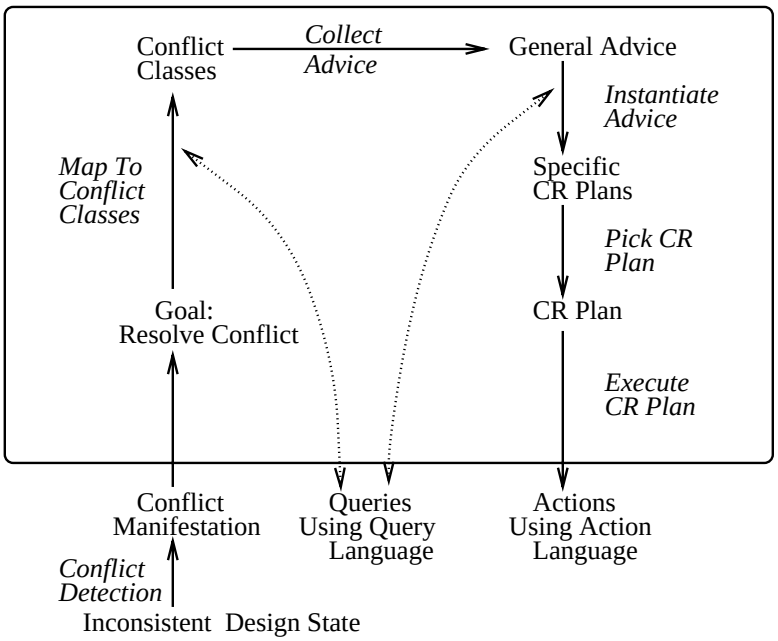

Figure 2.4: Klein's Operation of the Conflict Resolution Component

How the CR component resolves the conflicts is shown in Figure 2.4, and explained below. Once the conflict is detected, the conflict is mapped to the goal of resolving the conflict, and from there to a set of alternative CR plans for achieving the goal [Klein 1991].

When a conflict occurs, the conflict classes that subsume this conflict are identified. The general pieces of advice associated with these classes are then used as templates that are instantiated in the context of the conflict into specific conflict resolution plans. This is done by questioning the design agents using the *query language*. The plans accumulated by the instantiation process are then ordered using domain-independent heuristics, to find the one most likely to succeed. The top plan is then executed; the actions in this plan describe suggested design changes to the design agents using the *action language*.

Finding Conflict Classes

Classes in the conflict class taxonomy have abstract defining characteristics. The preconditions of these classes are expressed as a set of primitive question types that constitute the query language used by the CR component. The design agents are responsible for producing specific responses to these abstract questions. The CR component has in general an "abstract vocabulary" of abstract design entities (goal, plan, constraint, resource conduit, storage container) that the design agents should recognize.

Finding the conflict classes for a conflict involves determining which conflict classes have their preconditions satisfied by the conflict at hand. It is usually difficult to determine the cause for a conflict without having a complete model of the expertise in all the design agents involved in the conflict. As a result, the defining characteristics of a conflict class often have to be operationalized as a weaker set of conditions.

Collecting CR advice

Every conflict class has one or more associated abstract pieces of advice for resolving conflicts in that class. The conflict resolution advice applicable to a conflict are found by collecting all the pieces of advice associated with all the conflict classes that match the conflict.

Instantiating CR advice

The abstract pieces of advice that were collected have to be instantiated into specific CR plans suitable for the current conflict. The CR advice contains a number of "slots" that have to be filled with context-specific values in order to be executable. This is done by querying the design agents using the query language.

Selecting Conflict Resolution Plans

Once a set of CR plans for resolving the conflict are identified, the one that is most likely to succeed has to be selected. The CR component uses domain-independent heuristics to first suggest the CR plans that are more likely to succeed. Some of

the heuristics are, "choose the most specific plan available" and "choose the plan that makes the smallest changes to the design". The final decision concerning the best resulting design (CR plan), is left to the design agents since they are experts at evaluating design alternatives.

CR Plan Execution

When the conflict resolution plans are executed, suggestions are made to the design agents using the action language. The action language has proved adequate to express all the conflict resolution strategies considered so far.

When CR Plans Fail

Since conflict resolution expertise is heuristic in nature, a given conflict resolution plan may not work at all, or produce "secondary conflicts" as a result of trying to resolve the initial conflict. As a result it may take several CR suggestions to completely resolve a conflict. Conflicts resulting from suggestions of the CR component are treated the same as conflicts due to design agent actions. In such situations the CR component finds itself involved in the rationale for the conflict, will ask questions of itself using the query language, and may even give conflict resolution suggestions to itself using the action language. The CR component thus uses a single uniform mechanism for dealing with all types of conflicts.

2.3.6 Evaluation

Klein's implemented system, the cooperative design engine (CDE), currently creates designs for LAN's. It consists of machine based design agents having both the design and CR components. The different roles agents could take in the design process are, to refine a design or to critique an existing design from a particular design perspective. Design agents cooperate by refining and critiquing the component descriptions stored on a central blackboard. A domain-independent constraint propagation mechanism detects conflicts by looking for unsatisfiable constraints on a given component feature.

The design agents are implemented as rule-based expert systems, and there are six design agents with knowledge about Available LAN Technology, Security, Reliability, Vendor Needs, Expandability and Economics. The CR component includes a conflict class taxonomy with a total of 115 conflict classes [Klein 1991].

2.3.7 Summary

In Section 2.3, we saw Klein's model of [Klein 1990] run-time conflict resolution in cooperative design that is believed to have advantages over existing conflict resolution approaches [Brown 1985], [Lander et al 1991b]. It is based on the notions that conflict resolution expertise can be captured explicitly, and, in addition, can be organized usefully into a taxonomy of conflict classes and associated conflict resolution strategies. This theory is supported and instantiated in a study that examines conflict resolution among human experts in the domain of Solar Home design and LAN design. He concludes that design rationale is needed to support conflict resolution.

2.4 Lander's Research

2.4.1 Overview

In Section 2.4.2 we introduce Lander's Cooperating Experts Problem Solving Paradigm. In Section 2.4.3 we briefly describe her framework for cooperating experts, and in Section 2.4.4 we explain how she resolves the conflicts among these experts. In Section 2.4.5 we discuss how she coordinates the agents in the CEF framework. Finally in Section 2.4.6 we present how Lander evaluates her model in the domain of parametric design of steam condenser (STEAMER).

2.4.2 Cooperating Experts Problem Solving

Solving complex tasks require multiple specialized agents working together to achieve unified goals. To realize that potential, agents must have the capability to integrate their knowledge and skills productively. The paradigm of **Cooperating Experts Problem Solving (CEPS)** is a style of problem solving that allows

agents to communicate and cooperate through a shared language with shared expectations about how to reach agreement when conflicts occur. CEPS both preserves the autonomy of the agents and enables appropriate interactions [Lander et al 1991b].

Some examples of this integration of expertise through cooperation are seen in human problem-solving tasks such as design, research, business management and human relations. Consider a team of human experts, a manager and an accountant, who are cooperating to choose a telephone company for their office. Though they share the same goal of selecting an appropriate telephone company, each individual would like to insure that her own priorities receive top consideration. For example, the accountant would like to try Company X, because they are cost effective, while the manager would choose Company Y, because of their high quality service. Now if the office policy is to always choose the least expensive alternative when faced with this kind of conflict situation, then we have a global evaluation function which can be used to guide the decision. One of the motivating factors in the CEPS approach is the need to find a solution when there is no strong global model of correctness or optimality. This occurs when global evaluation criteria are absent, when global evaluation criteria are too expensive to compute, or when a global evaluation is some combination of a set of locally computed evaluations.

Why is Cooperating Experts problem solving a desirable paradigm? Bringing together diverse knowledge is a source of robustness and balance which is extremely important for many real-world problems. Experts working in a team can solve many problems that are beyond the scope of the individual experts. This also true for machine agents: integrating knowledge has the potential for increasing problem-solving power [Lander et al 1991b]. If we build agents that can work together even when individuals don't fully understand the entire task, we can begin to look at problems having a whole new level of complexity. This richness of combined knowledge and viewpoints from various experts provides the potential for creative problem solving and innovation.

Although diversity can be beneficial, there are difficulties with resolving the conflicts that arise when trying to merge multiple goals, priorities, and evaluation criteria for a common goal. Resolution occurs through the exchange of information among conflict participants. How to exchange, what to exchange, when to exchange and

with whom to exchange it are the questions that are addressed in the Cooperating Experts Framework (CEF) [Lander & Lesser 1989].

2.4.3 Cooperating Experts Framework

The **Cooperating Experts Framework (CEF)** is designed to support CEPS by providing the communication and conflict resolution structures and protocols required for cooperative interaction. The CEF framework supports diversity among agents, and agents need not be consistent in form, function or knowledge. The framework provides a kernel of procedures for communication, conflict resolution, scheduling and a language for shared information.

The architecture of the CEF framework is shown in Figure 2.5. The CEF control shell (Blackboard Monitor) integrates the execution of the framework knowledge sources and the agents in the agent set. The framework knowledge sources are executed by the framework knowledge source controller for performing high-level tasks on GLobal BlackBoards (GLBB) objects. The agent activation controller is used for execution of the independent agents.

When an agent enters the framework, it submits a list of interest areas to the blackboard monitor. Whenever a proposal is submitted in one of its interest areas, the agent is notified. Although CEF is implemented as a blackboard system, it does not use traditional blackboard knowledge sources as agents.

CEF is implemented in the Generic BlackBoards (GBB) framework as shown in Figure 2.5. The GLBB of CEF are used to facilitate communication among agents. Any information placed on these blackboards must be represented in a common language shared by all agents. This language is defined using the object definition capabilities of GBB. There are domain-independent, application-specific objects and blackboard structures. The domain-independent structures are supplied by the framework, e.g., a CONFLICT-OBJECT and CONFLICT-BLACKBOARD are available for any domain. Application (e.g. steam condenser design) knowledge is contained in the domain-specific objects and blackboard structures. In the steam-condenser design described in Section 2.4.6, examples of application-specific objects include a MOTOR-OBJECT, and a MOTOR-SPACE on the PROPOSAL-BLACKBOARD.

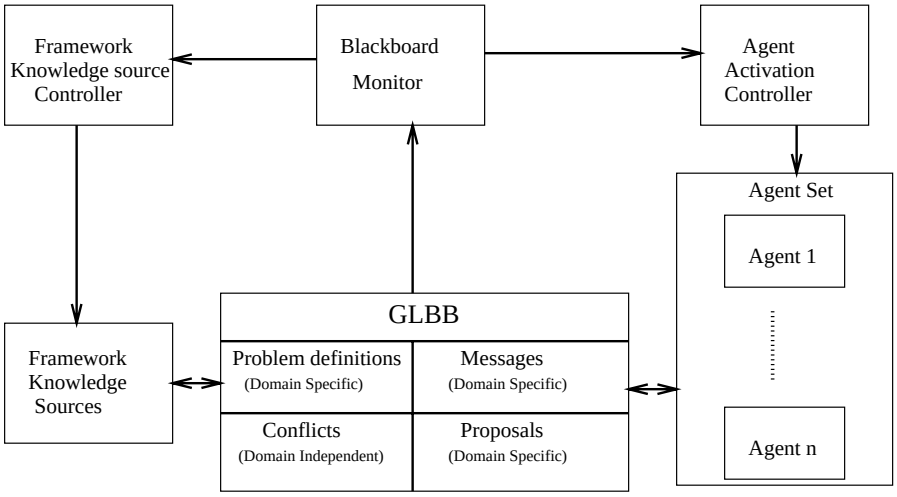

Figure 2.5: Lander's Cooperating Experts Framework (CEF) Architecture.

Table 2.1 shows the global blackboards (GLBB) for the steam condenser domain.

A CEF agent is a fully functional, autonomous and heterogeneous knowledge-based system which can solve problems in its limited domain independently. The agent does its own internal scheduling and has private data, knowledge, goals and history mechanisms. Although it can operate as a separate problem-solving entity, it has specific capabilities which allow it to act as a member of a team. Some of the capabilities which are provided by these agents include: a shared communication language; internal knowledge representations which capture sufficient goal and history information to allow for cooperative solution revision; mechanisms for incorporating externally produced partial solutions; mechanisms for negotiating conflicts; ability to coordinate an internal agenda with external events.

Within the agents, their evaluation criteria, constraints and goals must be explicitly represented since they are communicated to other systems as part of the conflict resolution process. Agents keep local histories of their actions and intermediate processing results to enable the revision of solutions in response to conflict situations. Agents local results are not accessible to other agents unless they are explicitly shared. If an agent's internal language is not the shared language, translation procedures must be provided for shared information. Solutions generated by CEF agents must be approved by all interested agents. Negotiating an agreement may require deferring

PROPOSALS BLACKBOARD
heat exchangers
motors
platforms
pumps
shafts
vbelts
condensers
problem definitions
MESSAGES BLACKBOARD
broadcast
framework agent
domain agents
heat exchanger agent
motor agent
platform agent
pump agent
shaft and vbelt agent
condenser agent
CONFLICTS BLACKBOARD

Table 2.1: Lander's GLobal Blackboards (GLBB) in STEAMER

or sacrificing some of an individual agent's goals. However, the negotiated solution takes into account all the relevant agents' constraints and preferences and satisfies the agents to whatever extent is possible.

2.4.4 Conflict Resolution

Conflicts are an inevitable and positive part of the problem-solving process in CEF. Conflict serves as a catalyst to exchange knowledge and goals among agents. According to Lander, conflicts could be either direct or indirect. For example, two agents may propose solutions for the same subproblem or for two subproblems which interact directly. On the other hand, two agents may propose solutions for subproblems that have no apparent relationship, but which ultimately interact through a chain of other subproblems.

Conflict sets are formed dynamically for each specific conflict situation by adding agents to the set as interactions between their subproblems are discovered. Once an agent has been incorporated into a conflict set, it will be notified of any changes that are suggested in response to the conflict. A framework KS does a cursory analysis of the conflicts at the global level. This analysis [Lander et al 1991b], includes the agents and parameters involved in the conflict and the depth of the solution path. This information is stored in a conflict representation object that is placed on the GLBB. Each of the agents are notified of the conflict situation. The agents now use the conflict analysis, and other globally available information such as proposals, local information about its own problem-solving resources, constraints, etc. They then decide what action to take.

When conflicts occur, the conflict participants must have protocols for the resolution process. This includes both a set of strategies for conflict resolution and a set of meta-strategies for choosing one. Lander [Lander et al 1991b], has specified a set of strategies which can be used to resolve conflicts. The choice of a strategy, given a particular conflict situation, is itself a knowledge-based problem. Information which can be applied to this choice includes, available problem-solving resources, the amount of effort that has already been expended in producing a solution, the solution's effectiveness, an estimate of the amount of processing required to generate a

new solution or to repair the current one, the dependency structure of related proposals, the importance of a particular component to the global solution, the number and type of conflicting parameters, the severity of the conflict and the flexibility of the agents involved in the conflict. Some strategies are computationally more expensive than others, some are inexpensive but less likely to produce promising proposals.

Some of the conflict resolution strategies used by Lander are Generate Random Alternatives, Compromise, Generate Constrained Alternatives, Generate Goal Alternatives, Case-Based Parameter Set Retrieval, and Revise and Merge Goals [Lander et al 1991b]. Conflict resolution protocols are realized as formal dialogues with specific actions that can be taken at each processing step. All agents know the protocols and can formulate the messages required for their role in a particular conflict situation.

2.4.5 Coordination in CEF Framework

According to Lander, for the agents in the CEF framework to function effectively, they must coordinate their internal activities with the global problem-solving situation. She describes the general coordination principles which treat the conflict resolution tasks as schedulable activities within a larger problem-solving context. These tasks are performed asynchronously by the systems involved in the resolution. She is still investigating various scheduling strategies, conflict resolution strategies and choice heuristics.

2.4.6 Evaluation

In order to evaluate the CEF framework, it is now being used as the basis for a system that does parametric design of steam condensers, STEAMER [Lander et al 1991b]. In parametric design, the general form of the artifact being designed is known, but the designer must find values for variable parameters of the artifact. The general form of a steam condenser consists of a pump, heat exchanger, motor, platform, shaft and v-belt. The CEF agents produce *proposals* which represent solutions to subproblems. In STEAMER, a proposal is either a component for a condenser or a complete condenser. Each component is designed by an agent with expert knowledge

and problem-solving methods that can include anything from numerical optimization to sets of heuristics.

STEAMER's agents correspond to the components of a condenser, e.g., the pump agent produces designs of pump components. There is also an agent that is concerned with characteristics of the complete condenser such as natural frequency, total weight and total cost. The components are independent except for shared parameters which represent the interface points of the design. The values for these parameters should be acceptable to all agents that use them for their processing. For example, the parameters of a pump component include water flow rate and power. The pump and heat-exchanger share the water flow rate parameter (water flows between pump and heat-exchanger) and the pump and motor components share power (the motor must deliver sufficient power to run the pump).

To begin processing, STEAMER is given a problem definition that specifies values for a set of condenser attributes. The problem definition is placed on STEAMER's global blackboard (GLBB) which is accessed by all expert agents and by a set of knowledge sources (KSs) which do high level operations on GLBB objects. The blackboards are shown in Table 2.1. Any agents that have work to do, whether pending or in reaction to new objects on GLBB, will execute during each processing cycle. In this case the pump, heat exchanger, and motor agents will begin local processing based on the new problem definition.

All agents in a CEF set must be able to perform a set of tasks such as: create new proposals, evaluate proposals, detect conflicts, and respond to conflicts. In response to the problem definition, each triggered agent performs a *create-proposal* task. As this is the starting condition, the problem-solving tends to be underconstrained. The agents work independently using whatever information is available and use default values when they must make assumptions about interface parameters. These assumptions are often unrealistic from a more global perspective, but the agents use this opportunity to put forth proposals that best reflect their own interests. It is a chance for each agent to act selfishly. As problem-solving progresses, it becomes more difficult for an agent to emphasize its own preference, so it is important to do so early on. Now, the agents' newly-generated component proposals are place on GLBB. KSs link the proposed components to a steam condenser proposal. Their compatibility

is checked by doing a syntactic analysis of the parameter names and values. When incompatibilities are found, the new proposals trigger *evaluation tasks* by any agents that share parameters with assigned values.

Evaluation tasks assign a local rating to a proposal. The agents use a shared rating scale and ratings have two components, compatibility and constraint satisfaction. The compatibility rating provides information about whether or not the proposal under evaluation is compatible with the current local situation. To determine a constraint satisfaction rating, an agent applies all relevant local constraints to the proposal under consideration. Constraints have attributes including flexibility, preferred and acceptable range of values, and importance. The degree to which constraints are satisfied provides the rating, which ranges from infeasible to excellent. Local ratings are attached to the proposal along with the evaluating agent's name.

Local ratings are combined into a global rating for the proposal by global KSs. Global ratings comprise of two attributes, acceptability and satisfaction. Acceptability is a function of constraint satisfaction, compatibility values from all agents, and system thresholds. For example, a possible function for acceptability is that all local ratings must be compatible, must have a constraint satisfaction rating of at least "fair", and the average constraint satisfaction rating must be at least "good". Satisfaction is a combination of the local constraint satisfaction values. Possible combination functions include average and minimum [Lander et al 1991b].

2.4.7 Summary

In Section 2.4, we provided Lander's basic framework to handle cooperative problem solving. We also discussed her technique of handling conflicts among the participating expert agents, and how they are resolved. Work is still in progress regarding the CEF framework and it is not complete.

2.5 Werkman's Research

2.5.1 Overview

In Section 2.5.2, we introduce Werkman's [Werkman 1992] Designer Fabricator Interpreter (DFI) Project, which is part of a comprehensive research effort intended to provide a distributed problem-solving environment. Section 2.5.3, explains how the DFI system evaluates a particular connection input and comes up with alternative connections. In Section 2.5.4, we briefly explain how Werkman's agents communicate by means of a centralized communication medium using speech acts. Section 2.5.5, explains the role of the arbitrator in the negotiation process. In Section 2.5.6 the Negotiation Scheme used in the DFI system is illustrated and in the last Section a sample evaluation is shown.

2.5.2 Designer Fabricator Interpreter (DFI) System

One of the objectives of the DFI Research Project is to develop a computer tool that allows structural design engineers to get different opinions from multiple view points about beam-to-column connections in buildings. Unlike many civil engineering design knowledge-based systems which attempt to optimize structural design based on one aspect, e.g., minimum steel weight, the DFI system attempts to develop a framework for distributed construction agents. The agents interact and present their different view points on multiple aspects about beam-to-column connections. The DFI system reflects the distributed nature of the construction industry by providing a multi-agent architecture which models design, fabrication and erection processes. The system considers issues that are important to each participant in the design process and produces a cooperative solution, through negotiation. In addition, representing specialized construction process knowledge as agents permits easier testing and maintenance as new knowledge is acquired. Finally, the modular nature of the architecture permits the addition of new agents with new construction expertise in a straightforward fashion.

The agents in the DFI system behave in both a cooperative and competitive fashion. When they work towards a common goal by suggesting alternative connections

to the one originally specified by the user, they exhibit cooperative behavior. They also behave competitively during the proposal process by maximally improving their own positions. To provide some means of balance, an independent arbitrator agent is used to monitor the agent proposal process. The arbitrator mediates during the agents proposal process by using an abstract level of shared knowledge about each agent's issues. The final set of alternative connections produced by the negotiated evaluation process is considered generally acceptable by all agents.

2.5.3 DFI Evaluation Process

The DFI system evaluates and suggests alternative connection designs based on multiple agent viewpoints. Each agents viewpoint is further decomposed into several unique issues. The issues are based on different aspects of connections such as economics, feasibility, type of material, etc, as shown in Table 2.2. The importance of an agent issue depends on which agent viewpoint one takes, i.e., designer, fabricator or erector, within the context of a specific connection evaluation. During the proposal process, an agent will look at each connection previously proposed by other agents and evaluate any affected issues.

Agent Issues	Designer Agent	Fabricator Agent	Erector Agent
Strength	3		
Stiffness	3		
Reliability	2		
Versatility	2		
FAB. Cost		4	
FAB. Ease		3	
MAT. Cost		2	
ERE. Cost			2
ERE. Ease			2
Safety			3
Composite Score	2.75	3.67	2.33

Table 2.2: Werkman's Example Rating Factors

The DFI system requires the user to select a connection from the building description initially entered into the system. The user is then required to enter a "key issue" (e.g., strength, fabrication cost, safety), which forces the agents to focus their evaluation of a connection. Once this initial information is provided, the arbitrator takes control and selects the agent (design, fabrication or erection) worst affected by the user's initial connection. This is done by taking a composite or average score of each agents issues. The agent with the lowest composite score gets to evaluate the connection first. Table 2.2, shows the Rating Factors Table with the composite score computed for each agent. For the case shown in the table, the Erector would evaluate the connection first with a composite score of 2.33 while the Designer and Fabricator have composite scores of 2.75 and 3.67 respectively.

The evaluating agent then selects the worst issue and attempts to improve it by suggesting alternative connection configurations. Prior to selecting an alternative configuration, the evaluating agent must search its connection database and select all of the connections which have a greater value on the agent's "worst issue" and also a higher value than the original one for the "key issue" provided by the user. Once this set of connections is determined, the evaluating agent takes the composite score of all the connections in the set and selects the configuration with the highest value. This connection is then posted to a centralized communication area along with additional information about other possible alternatives. This additional alternative connection information is used later by the arbitrator when an agent requires help in suggesting an adequate alternative.

2.5.4 DFI Agents Communication

Initially the user selects the connection (between a particular column and beam) from the description of the building. Calculations are performed to determine the moment capacity of the beam and establish the required connection type. The system then provides the user with a list of connection component alternatives. Now the user can evaluate the connection design to see what effect it has on the fabrication and the erection process. After the multi-agent evaluation has been completed, the user is presented with the original connection configuration and the three potentially different

connection configurations proposed by the design, fabrication and erection agents. The user has the final choice in selecting the connection type from the available choices.

During the multi-agent evaluation process, agents communicate by means of a centralized communications area called the blackboard [Nii 1986b]. The blackboard scheme allows agents to post messages (to who, from who, message content) as well as read messages from other agents. The use of such a scheme allows the system to maintain a history of the agents' dialog as the proposal and negotiation process proceeds. This scheme, combined with a common language (primitive interagent messages), allows for an effective form of negotiation between agents which allows them to reason about the beliefs of other agents.

This interagent language, based on speech act theory [Austin 1962] [Searle 1969], allows for expression of agent intentions at some level of abstraction. Each social action that an agent might enter into contains a case structure (to, from, action, etc.), preconditions (necessary agent conditions), and postconditions (results of successfully performing the action). The speech acts based communication provides for a plan-based approach for exchanging information. This plan-based approach means, that once an agent receives a specific message from a sending agent, the receiving agent will know the type and form of response message with which it is expected to reply. This makes for short explicit messages and reduces extraneous message overhead.

2.5.5 Arbitrator

The arbitrator agent resides at the logical center of the agent interaction process, monitoring all agent communication, allowing the arbitrator to assist in the problem-solving process when necessary. A *DFI Relational Network* [Werkman 1992] is maintained to represent the semantic relationships between the connection aspects and agent issues. This network shows how a connection is related to designer, fabricator, and erector agents "issues" in terms of the functional, component and fast-ner "aspects" of the connection. This network only provides the arbitrator with an abstract level description of issues and how they relate to one another within the connection domain. This provides the arbitrator with enough information to detect interagent

issue conflicts and assist them in the negotiation process. The arbitrator does not contain any knowledge about each agents unique operational knowledge. In order for the arbitrator to arbitrate a proposed solution it has to query the agents and find out the reason and explanation behind the issue relationship under consideration. This network scheme allows for the addition of new agents to the distributed problem solving model. Initially each agent must share its knowledge of relevant issues with the arbitrator so that they can be added to the network and used during negotiation.

2.5.6 DFI Negotiation Scheme

A negotiation scheme has been devised for distributed problem-solving that takes place between the semi-autonomous agents in the DFI system. During the connection evaluation process, agents' comment on connection characteristics based on their unique set of issues. As shown in Figure 2.6, each *reviewing agent* determines its best possible alternative connection configurations while maintaining or improving the value of the users initial key issue.

During the evaluation process, the *reviewing agent* evaluates the *proposing agent's* connection and determines which issue is most problematic. The *reviewing agent* then generates alternatives that enhance the worst issue and submits this list for review to the *key issue agent*. The *key issue agent* (containing the user's "key issue") selects only the connections that meet or exceed the key issue and then returns this new list to the *reviewing agent*. In an attempt to provide a cooperative solution, the *reviewing agent* sends the list of alternatives which meets the key issue to the *proposing agent* for review. This gives the *proposing agent* a chance to order the list of alternatives based on the *proposing agent's* preferences. The *reviewing agent* then takes this ordered list and selects its best possible connection counterproposal in response to the *proposing agent's* initial proposal. The *reviewing agent* saves this specific counterproposal and performs a similar evaluation for all other agents in the system. Upon evaluating all other agent proposals, the *reviewing agent* then selects the best counterproposal from all agent specific counterproposals and proposes that connection as its response to all other agent connections currently proposed on the blackboard.

During the agent evaluation and negotiation process, a *proposing agent* might

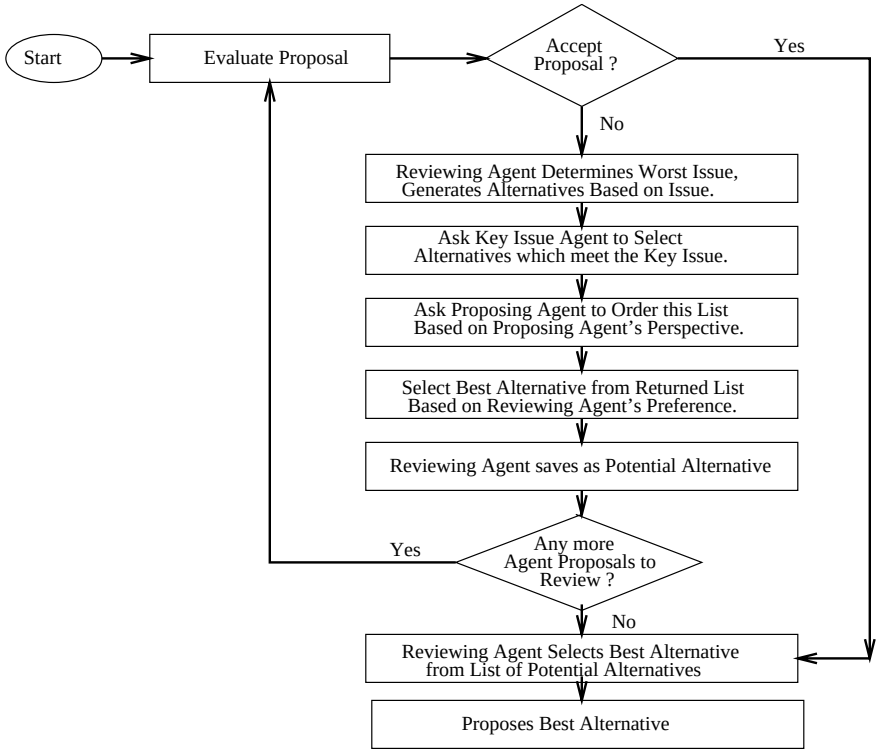

Figure 2.6: Werkman's Evaluation and Proposal Process

exceed the acceptable limits of the issues (values) of the group. This may require an agent to concede an issue and propose an alternative in order for the negotiation to proceed. It is also possible that an agent may not be able to concede an issue because it would be too costly for the agent. In such cases, the arbitrator agent must be brought in to attempt to mediate a solution between the two conflicting agents. Initially, the arbitrator monitors the current status of all agent proposals and reviews each proposal for any immediate problems it might cause for an agent. If the arbitrator detects a problem that affects a particular agent, it warns the agent and gives control to that agent so that it has a chance to respond to the problem caused by the proposal connection.

In addition to detecting agent problems during proposals, the arbitrator also reviews the history of proposed connections to determine if a "halting" condition or "deadlock" situation has occurred. When two agents propose the same connection, the arbitrator detects this and informs all agents that the evaluation process has come to a halt. The resulting connections are presented to the user for review. If the arbitrator notices that the same proposals are being made by the same agents in response to a previous agent's proposal, then a "deadlock" situation has occurred. The arbitrator intervenes by analyzing the situation and attempts to convince one agent that the other would agree if only the first agent would relax the importance of an issue or drop it altogether. The arbitrator generates the argument of which issues are relevant for an agent from the abstract interagent issue relations maintained by the DFI relational network as well as the history of past proposals and issues. In situations where agents still fail to agree after initial negotiation methods, the arbitrator determines the final solution given the input from both agents as to the importance of each agent's issue. This is a form of meta-level control, in which the final decision is based on an *a priori* policy of acceptance, specific to the given domain (construction). If the agent's proposals do not converge after six iterations, the arbitrator stops the evaluation and returns control to the user.

The agents are generally executed in a pre-determined default order, if the arbitrator sees no problems. The actual order of agent proposals is determined by the arbitrator, when appropriate, by using knowledge of the agent's issues and connection rating values. This scheme allows DFI to take an approach that uses aspects of both

centralized control (arbitrator) as well as agent based control over negotiation.

2.5.7 Evaluation

In this section we will present an example of a connection evaluation with negotiated alternative proposals between the agents. Once the user enters a connection, he is asked to enter the "key issue", which is maintained by all agents during their proposal of alternate connection configuration. In the example illustrated, the user specifies an *endplate* connection with a key issue of *strength*. The agents will now attempt to suggest alternative connections that are of the same connection (endplate) type and with the same or higher value for key issue (strength).

Initially the arbitrator *commands* the design agent to *accept* the user's endplate connection proposal using strength as the positive supporting issue, because the user is the designer in the first cycle of negotiation. The design agent then informs all agents of the key issue and request that the proposed connection be evaluated. Before each agent's evaluation, the arbitrator reviews all proposed connections to determine which agent is most detrimentally affected and hence should go next. In this case, the erector is most severely affected by the designer's endplate proposal. The erector determines that the designer's proposal is unacceptable because the endplate connection is too difficult in terms of erection ease. Therefore, the erector *refuses* the designer's connection and looks to the fabricator in hopes that it might have proposed an acceptable connection. At this stage the fabricator has not yet proposed anything, so the erector selects a connection from the set of possible connections about which it knows. The erector *requests* the plates-tee because it satisfies the erection ease issue as well as satisfying the user specified key issue.

Now the erector directs the proposed connection back to the designer for review. The designer *accepts* the erector's proposed connection because it exceeds the key issue of strength as well as meets the designer's criterion for the endplate connection. Also the value of the key issue has been increased to the new value associated with the erector's proposed plates-tee connection since it was higher than the original designer's strength key issue for the endplate connection. By increasing the value of the key issue, the search space of possible connection alternatives is reduced, thus

causing the agents to converge more quickly on a set of agreeable connections.

Next, the arbitrator reviews the connection situation and notices that the two agents have proposed the same connection. Generally, this would cause the arbitrator to *inform* all agents of a halting condition. This is not the case here because, the fabricator has not yet had a chance to evaluate any connections. Thus, the arbitrator gives control to the fabricator who looks at the designer's connection and immediately notices that material cost is the problem issue for it. Since both the designer's and erector's connection are the same, the fabricator needs only to review the plates-tee connection and propose an alternative connection. The next best connection that maintains the key issue of strength as well as improves the fabricator's material cost is the direct flange weld with shear plate.

Again, the arbitrator reviews the evaluation process and notices that the two agents have agreed on a connection and that each agent has had a chance at suggesting an alternative. The arbitrator *informs* all agents of the halting condition and control is returned to the user. At this point the user can ask any agent to explain its proposed connection or continue with the evaluation. If the user continues, the arbitrator reviews the situation and notices that no particular agent is in "trouble" and allows the agents to determine their own control sequence. Whichever agent received the last message is given a chance to respond to that message. In this case, the fabricator proposed a connection to the designer. The design agent, upon reviewing this connection notices that the fabricator's connection satisfies all issue of the designer. Thus, the design agent accepts the fabricator's proposal of direct flange weld with shear plate.

2.5.8 Summary

As explained in Section 2.5, the Designer Fabricator Interpreter (DFI) system is a step towards providing a distributed problem solving environment which allows semi-autonomous agents to work cooperatively by reasoning and negotiating about the current problem based on their expert viewpoints as well as the viewpoints of other agents. An arbitrator agent assists in the negotiation process when agents cannot come to an agreement. All communication is coordinated through shared knowledge

model and a set of communication primitives.

2.6 Kannapan's Research

2.6.1 Overview

In Section 2.6.2 we discuss Kannapan's views on concurrent engineering. In Section 2.6.3 we briefly describe the utility functions on which his negotiation model is based. Section 2.6.4 explains his negotiation model and in Section 2.6.5 we outline his negotiation protocol. In Section 2.6.6 we present his evaluation of his negotiation model.

2.6.2 Concurrent Engineering

According to Kannapan, in a typical concurrent engineering environment, human designers specializing in different fields interact through a common communication medium. In an automated CE environment he views the same scenario where, intelligent design agents work in a team, in parallel and share design information. The design agents suggest, critique and implement changes to the product design. The incremental decisions made by the design agents represent the evolution of a design, incorporating the concerns arising throughout the product life cycle [Kannapan & Marshek 1991].

2.6.3 Utility Functions

Kannapan's negotiation model is based on utility functions. Utility functions define the utility value of a particular parameter. There are several ways in which utility functions for parameters can be obtained. The functions can be constructed by directly questioning the decision maker, from domain specific design knowledge, from experimentation and engineering principles, or retrieved from previous similar cases. For his work, the utility functions derived from handbook design knowledge and engineering principles, are simplified and approximated to piecewise linear functions. Because of the subjective elements in the utility function calculations, different agents may have different utility functions for the same design parameters [Sycara 1991].

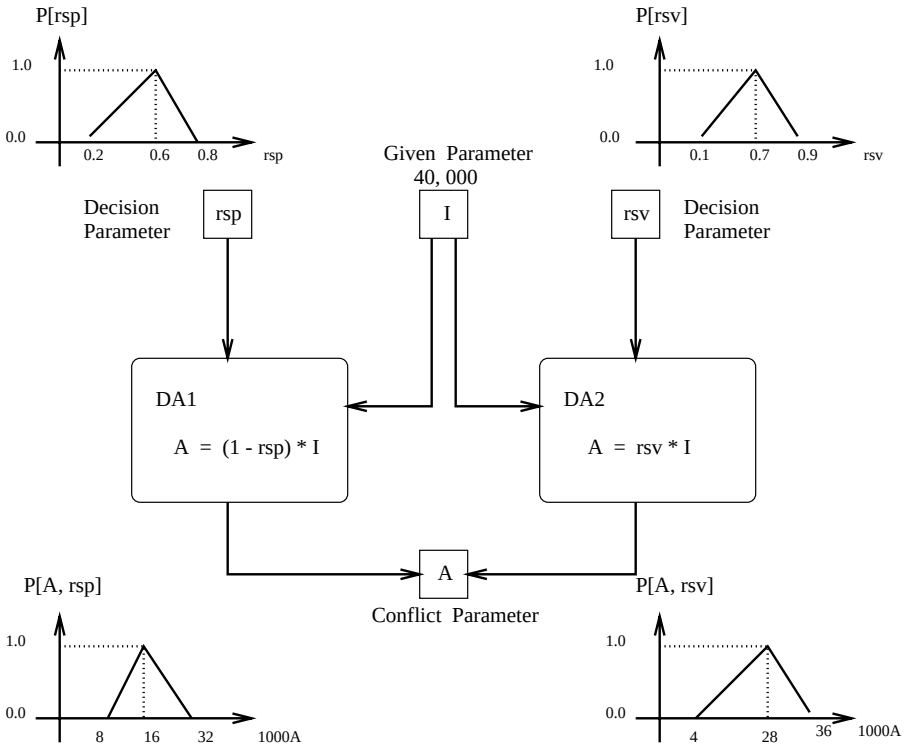

Figure 2.7: Kannapan's Utility functions and their propagation

To illustrate the utility functions, Kannapan [Kannapan & Marshek 1991], uses two agents DA1 and DA2 who have to decide the amount of money (A) to be put in a certain investment based on their total income (I). DA1 decides on the ratio (rsp) of income to be spent, and DA2 decides on the ration (rsv) of the income to be saved. Now the utility functions for decisions on values for rsp and rsv by agents DA1 and DA2 respectively are defined as shown in Figure 2.7. Utility functions P[rsp] and P[rsv] are defined between say 0.0 (least preferred) and 1.0 (most preferred) for values of the parameters controlled by each agent as shown next to the parameters rsp and rsv in Figure 2.7. The values and corresponding utility levels for rsp and rsv are propagated through the computations to get the utility functions for A from the point of view of each agent as shown next to the parameter A in Figure 2.7.

2.6.4 Negotiation

In his concurrent engineering scheme described in Section 2.6.2, there are three types of parameters, namely *decision parameters, given parameters,* and *conflict parameters. Decision parameters* are those parameters on which the design agents are allowed to make decisions independently and in their control. *Given parameters* are those parameters specified in the design requirements, among which some are exclusive and some are shared. *Conflict parameters*, are those parameters that arise due to the propagation of given (shared) parameters and decision parameters through engineering relationships between these parameters. Resolution of the conflict in values of shared parameters require some form of negotiation between the design agents to agree on mutually acceptable values. The model of negotiation discussed here is based on the classical model of utility and economic negotiation.

These different types of parameters, such as given parameters, decision parameters and conflict parameters defined above are explained through an example. He shows how the Nash and Kalai-Smorodinsky solutions for negotiation can be applied to conflict situations to resolve conflicts. The Nash solution maximizes the product of utilities of the design agents, while the Kalai-Somorodinsky solution maximizes the utilities of each design agent while requiring that each design agent achieve the same proportion of the maximum utility it is capable of achieving. The Nash

and Kalai-Somorodinsky solutions are obtained from axiomatic theories of bargaining/negotiation without threats.

2.6.5 Negotiation Protocol

Kannapan describes the protocol for negotiation and resolution of a conflict between intelligent design agents on the value of a shared parameter as follows:

1. Determine the set of exclusive parameters controlled by each of the design agents, i.e. decision parameters that affect the conflict parameter.

2. From the sets of parameters determined in Step 1, select one decision parameter per design agent that affect no other conflict parameter; propagate the utility functions of the selected decision parameters to the conflict parameter, assuming all other parameter values to be unchanged.

3. Compute the Nash/Kalai-Somorodinsky negotiation solutions for the conflict parameter from the utility functions of the conflict parameter obtained in 2.

4. Propagate the negotiated solution of the conflict parameter back to the design agents to determine the negotiated values of the decision parameters selected in 2.

This protocol is suitable for parameter relationships that are explicit and under-constrained. If the relationships are not explicit, parameter values cannot be propagated forward and backward as required in Step 2 and Step 4. If the relationships are over-constrained there will be no decisions required to set values for exclusive parameter.

2.6.6 Evaluation

Kannapan has selected the concurrent engineering design of a poppet relief valve to illustrate the process of negotiation and resolution of parameter value conflicts. The intelligent design agents (DA's) involved in the design of the poppet relief valve are shown in Figure 2.8. They include a Valve DA comprising of a Valve-flow DA and a Valve-cracking DA, a Helical-spring DA and a Pipe-enclosure DA. During the design, the Valve-flow DA executes first, the Pipe-enclosure DA executes in parallel with the Valve-cracking DA and the Helical-spring DA. The Valve-cracking DA executes before the Helical-spring DA. The given parameters, decision parameters,

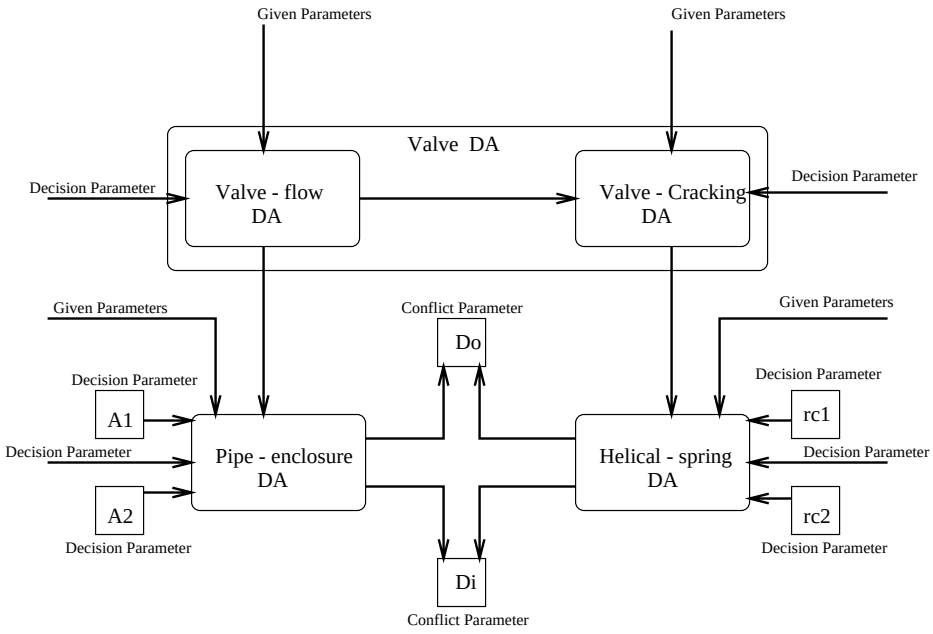

Figure 2.8: Kannapan's Concurrent design diagram for poppet relief valve showing conflict in parameters Di and Do.

and conflict parameters are also labeled in Figure 2.8. The parameters controlled by each design agent are identified from standard procedures for poppet valve and pipe design available in literature. Values of given parameters and decision parameters for a representative design are shown alongside the parameters in Figure 2.8. The individual decisions made by the design agents in this case lead to conflicts in values for the parameters D_i and D_o. By using the protocol outlined in Section 2.6.5, we will see how these conflicts could be resolved.

By executing Step 1 of the negotiation protocol, a negotiation graph is constructed for the conflict parameters D_i and D_o. The negotiation graphs for D_i and D_o look similar to Figure 2.7. Given parameters, decision parameters and relationships that result in a conflict parameter are nodes in a negotiation graph. Directed edges show the propagation of the parameter values. The decision parameters selected in Step 2 of the protocol are A_2 and r_{c2} for D_i, and A_1 and r_{c1} for D_o. A_1 and A_2 are the corrosion allowances for the pipe enclosure and the poppet stem respectively with values decided by the Pipe-enclosure DA. The r_{c1} and r_{c2} parameters are clearance

ratios on the outside and inside of the helical spring with values decided by the Helical-spring DA. The utility functions for the decision parameters A_2 and r_{c2} are P$[A_2]$ and P$[r_{c2}]$ respectively. The utility functions for A_2 and r_{c2} are propagated to D_i. The propagated utility functions of D_i from the point of view of each of the design agents are computed, i.e. P$[D_i, A_2]$ is the utility function of D_i propagated from the utility function of decision parameter A_2. The Nash and Kalai-Somorodinsky solution for the conflict parameter D_i is computed as described in Step 3. Similar steps are followed for computing the value of the other conflict parameter D_o. Then the values of the decision parameters are obtained by Step 4 of the protocol by using the Nash/Kalai-Somorodinsky solutions for the conflict parameters.

2.6.7 Summary

In Section 2.6, we saw how Kannapan resolves conflicts by using the Nash and Kalai-Somordinsky solutions developed in the literature for industrial and social negotiation/bargaining. Unlike iterative approaches to the negotiation process, he specifically uses the utility function based model which proposes negotiated solutions without iteration.

2.7 Other Related Research

There are many other researchers who have worked in the area of negotiation and conflict resolution. Their work has been referenced in [Bond & Gasser 1992] and some of these papers include [Velthuijsen & Griffeth 1992], [Zlotkin & Rosenschein 1992], and [Rosenschein & Genesereth 1985].

2.8 Summary

This chapter has presented summaries of some of the current research in the area of negotiation and conflict resolution. This thesis does not constitute a comprehensive view of these issues, rather it presents those which were considered in developing the I3D+ Negotiation System. The next chapter presents a comparative analysis of the various negotiation systems we described in this chapter.

Chapter 3

A Comparative Analysis of Negotiation Architecture

3.1 Overview

This chapter presents the comparative analysis of negotiation systems, specifically the negotiation architecture. In Section 3.2, the main attributes of negotiation that characterizes any negotiation system are explained. In Section 3.3, these attributes are used to compare the selected negotiation systems [Sycara 1991], [Werkman 1990], [Lander et al 1991b], [Klein 1991], and [Kannapan & Marshek 1991].

3.2 Attributes of Negotiation

3.2.1 The Computational Model

The most important attribute to characterize the negotiation process is the kind of computational model researchers used. The model that is used to automate the negotiation process is referred to as the computational model. The computational model differs from system to system. Some of the computational models that have been implemented are based on Case-Based Reasoning and Preference Analysis [Sycara 1991],

Utility Functions [Kannapan & Marshek 1991], Conflict Classes [Klein 1991], etc. We will compare and analyze each of these models in Section 3.3.

The computational models are broadly classified by Klein into three categories [Klein 1991]. In the first category, there is no need for negotiation because all the potential conflicts are "compiled" out by virtue of exhaustive discussions when the system is developed. This category is called *Development-Time Conflict Resolution.* In the second category, the system allows conflicts to be asserted by the design agents as the system runs, and then resolved by some kind of conflict resolution mechanism. Some of the approaches used in this category are backtracking, numerically-weighted constraint relaxation and pieces of specific conflict resolution advice [Brown 1985]. This category is called *Knowledge-Poor Run-Time Conflict Resolution.* In the third category, conflicts could happen at any time during the design process and there is a separate conflict resolution component available to handle the conflicts. This category is called *General Conflict Resolution.* In this category, conflict resolution is given first class status, in the sense that is it represented explicitly and completely separated from the domain knowledge.

3.2.2 Central Monitor/Arbitrator

The second major attribute is the presence or absence of a central monitor to monitor the negotiation process. This central monitor also serves as an arbitrator in some systems. The major role of the central monitor is to oversee the negotiation process and check for halting conditions or potential deadlocks. In case of a deadlock the central monitor either tries to resolve the deadlock or notify the user of such a situation. When the central monitor performs the role of an arbitrator, it has powers to resolve the deadlocks without user intervention.

The monitor may also allow a certain number of iterations for the agents to resolve their conflicts, typically around 6 iterations. If the agents fail to resolve their conflicts within the given time frame then the monitor takes control and takes appropriate action. The presence/absence of the central monitor also to a certain extent dictates the kind of negotiation strategies that are being followed. We will discuss the negotiation strategies later (Section 3.2.7).

3.2.3 Conflict Detection

Some of the other attributes of negotiation are *conflict detection, conflict notification* and *conflict resolution*. Some systems perform each of these functions explicitly. Other systems assume that *conflict detection and notification* has already been done, (i.e, that there exists a conflict situation) and goes ahead with *conflict resolution*.

The *conflict detection* mechanism detects conflicts. The way conflicts are detected, however, are dependent on how each individual system implements it. The conflicts could be classified into two major categories:

- two agents try to decide the value of a single parameter.
- one agent has negative criticism about the decision made by another agent.

In the first category, the second agent which tries to update the original value, will detect the conflict. In the second category, when the critic agent is scheduled to execute, it detects the conflict, if it does not like the original value.

3.2.4 Conflict Notification

Another attribute of negotiation is *conflict notification*. Generally, in systems where there is a central monitor, conflict notification is done by the central monitor to the agents involved in the conflict. In the absence of central control, conflicts are detected by each individual agent, and the agent detecting the conflict notifies the other agents involved in the conflict, about the conflict situation. Hence, in the case where there is no central monitor, the individual agents are responsible for conflict notification.

To a certain extent, the function of conflict notification is decided by what kind of a negotiation architecture is being followed. In systems where there is centralized control, i.e., systems with a central monitor, the monitor takes care of conflict notification. In systems where there is a distributed approach, i.e., systems where there is no central control, the individual agents detecting conflicts are responsible for conflict notification.

3.2.5 Conflict Resolution

The other important attribute of Negotiation is the process of *conflict resolution*. This attribute is closely tied to the *negotiation mechanism* explained in Section 3.2.8. The conflict resolution process is the implementation of the computational model explained in Section 3.2.1. Once the conflicts are detected and notified, the conflict resolution process takes over and resolves the conflicts.

The process in which these conflicts are resolved varies from system to system. Some of the typical conflict resolution processes are based on Case-Based Reasoning and Preference Analysis [Sycara 1991], Utility Functions [Kannapan & Marshek 1991], Conflict Classes [Klein 1991], etc. We will compare and analyze each of these systems in Section 3.3.

3.2.6 Use of Design Rationale

Another attribute of negotiation is the presence/absence of a Design Rationale/History. Some systems just maintain a design history, i.e., a list of design decisions in chronological order, to refer back to during the negotiation process. Some systems, along with maintaining a design history, also maintain the justifications/reasons for making these design decisions. This more complete form of design history is called the design rationale. Systems make use of the design rationale, to avoid backtracking to some extent, because they avoid using the same design decisions that have been tried earlier and proved to be futile. Design rationale can also be used by the negotiation process for making efficient negotiation decisions.

3.2.7 Negotiation Strategies

The other major attribute of negotiation, is the kind of *negotiation strategies* that could be followed. The various strategies could be either pre-determined or dynamically adapted to the situation. Some strategies could have a centralized approach, with a central monitor controlling the negotiation process, or a distributed approach in which each individual agent controls the negotiation process.

A centralized approach allows the agents to be semi-autonomous. In this case,

the agent architecture could be simple and easy to implement. This is because the agents are not expected to know with whom they have to negotiate. The central controller has all the necessary information to coordinate the negotiation process. This centralized approach reduces some amount of flexibility, because the agents cannot carry on the negotiation process on their own.

In the distributed approach, the first visible thing is the absence of a central monitor. Agents here are autonomous and interact directly with each other to resolve their conflicts. The agent architecture is also complicated, because they all should have the knowledge required to handle the negotiation process.

In the hybrid approach we have a mixture of both these, in which there is a central component and also the agents have enough autonomy to interact on their own.

Sycara [1989] has a very distributed architecture, where as Klein [1991] has a centralized approach. Werkman [1992] has a combination of both these approaches. We will compare and analyze each of these approaches in Section 3.3.

3.2.8 Negotiation Mechanism

Another attribute of negotiation is the *negotiation mechanism*. The negotiation mechanism is closely tied down to the *conflict resolution* process explained in Section 3.2.5. The negotiation mechanism differs from system to system and each system has a unique way of implementing it.

In some systems the negotiation mechanism is based on sending proposals and justifications, receiving evaluations of the proposals, and, if needed, arguments and justifications for changing them. For example, [Sycara 1991] uses Cased-Based Reasoning (CBR) and Preference Analysis for proposal generation, proposal evaluation, proposal modification and argumentation. Some systems use Utility Functions [Kannapan & Marshek 1991] to propagate the necessary parameters (constraints) back and forth and to resolve these conflicts. Some systems use conflict classes [Klein 1991] and their associated strategies, to resolve conflicts. We will compare each of these mechanisms in Section 3.3.

3.2.9 Evaluation

The last attribute of negotiation is how the negotiation process has been evaluated. Is there an implementation of their negotiation model, or is the process explained with an example. The evaluation process differs from system to system, because some systems just give an example to explain their negotiation model, while others implement their negotiation model and demonstrate its working with a sample run.

3.3 Comparison of Negotiation Architectures

3.3.1 Sycara's Negotiation Architecture

Comparison of Sycara's Negotiation Architecture with the attributes of negotiation are presented in Table 3.1. Each of these attributes are briefly discussed below.

In Sycara's Negotiation Architecture, the computational model is based on the integration of case-based reasoning, qualitative reasoning, constraint relaxation and comparison of utilities. Case-based reasoning is used to propose "ball-park" solutions, to also refine such proposed solutions and compromises, while reasoning with utilities is used in situations where case-based reasoning is not appropriate.

Her model captures the dynamic interactions of the cooperating agents during negotiation. There is no central monitor or arbitrator present in her architecture. The agents are capable of communicating directly with each other and resolving the conflicts. In contrast to other architectures [Klein 1990], [Kannapan & Marshek 1991], [Werkman 1992], where different agents propose values for a design attribute and a central arbitrator evaluates and selects among the proposed values, i.e. there is no interaction among the agents, her model captures the full complexity and dynamics of interaction between the different agents.

Sycara claims that existing work on CE (e.g., [Londoño et al 1991]) has focused on the communication architecture and conflict detection between agents, but nothing much has been done on modeling the negotiation process. Hence, her work assumes that there is a conflict detection and conflict notification mechanism already available. She focuses mainly on modeling the process of reconciling design decisions and design proposals that arise from the different agents during the design process in order to

form an acceptable compromise.

In her negotiation architecture, the input to the negotiation process is the set of conflicting goals and violated constraints of the various design agents. The final output is either a resolved, single consistent set of design decisions, or an indication of failure. The negotiation mechanism consists of proposal generation, justification and critiquing of the proposal, and repair and improvement of a rejected proposal.

Attributes	**SYCARA'S Model**
Computational Model	Integrates CBR, Multi-Attribute Utilities & Constraint Relaxation
Central Monitor	NONE
Conflict Detection	Beyond scope of their work
Conflict Notification	NONE
Conflict Resolution	Done through proposal generation, feedback, modification and justification
Use of Design Rationale	Leads to major efficiency gains and reduces backtracking
Negotiation Mechanism	Iterative and done through proposal generation, feedback, modification and justification
Negotiation Strategy	No centralized control, direct communication between agents
Evaluation Criteria	Design of Turbine Blade (Not yet implemented)

Table 3.1: Comparison of the Attributes of Negotiation - Sycara's Model

Sycara maintains a design record, in which the descriptions of the problem specifications in terms of design goals and constraints, the solution of the design problem, and the trace of the decisions that show why the solution satisfies the problem spec-

ifications are kept. This record helps to reduce backtracking during the negotiation process and achieve major efficiency gains. This is done by referring back to the record in future negotiations, and avoiding any potential cases for which the negotiation has failed earlier.

She mentions that her system is not fully implemented [Sycara 1991], but has given an example, the process of designing a turbine blade. Some of the dominant agents involved in this example are the aerodynamics agents, the structural engineering agent, the manufacturing agent and the marketing agent.

3.3.2 Werkman's Negotiation Architecture

Comparison of Werkman's Negotiation Architecture with the attributes of negotiation are presented in Table 3.2. Each of these attributes are briefly discussed below.

Werkman uses Knowledge-Based Expert Systems for his Negotiation Model. In this model, the agents communicate their issues about the problem, reason from their own and other agent's perspectives, and finally generate acceptable counterproposals. In Werkman's architecture, if the agents cannot resolve their conflicts, then an arbitrator agent is involved to help the conflicting agents reach an agreement. Hence his architecture is centralized and the arbitrator agent is used to monitor and mediate the agent proposal process. The arbitrator agent resides at the logical center of the agent interaction process, monitoring all agent communication, allowing the arbitrator to assist in the problem-solving process when necessary.

The central arbitrator detects the conflicts among agents and also notifies them. The arbitrator is also responsible for detecting the halting condition once the negotiation process is initiated between the agents. If the agent's proposal do not converge after six iterations, the arbitrator stops the negotiation and returns control to the user.

The actual order of executing agents is determined by the arbitrator when appropriate. If the arbitrator "sees" no problems, then the agents follow a predetermined default order. This scheme allows an approach to negotiation that uses aspects of both centralized control as well as agent based control over negotiation. This negotiation

strategy differs from systems which contain fully autonomous agent control schemes where agents are totally on their own to determine what to do next [Sycara 1991], [Lander et al 1991b] and centrally controlled systems where one superagent maintains total control over all other agents [Klein 1991].

Attributes	**WERKMAN'S Model**
Computational Model	Knowledge-Based
Central Arbitrator	Controls & Selects Agents, develops alternatives during conflict resolution
Conflict Detection	Arbitrator detects interagent issue conflicts
Conflict Notification	Done by Arbitrator
Conflict Resolution	Done through issue unlinking and allows agents to consider viable alternatives
Use of Design Rationale	Agent dialog is maintained for later negotiation purposes
Negotiation Mechanism	Iterative and done through Issue relaxation, and interagent proposal generation
Negotiation Strategy	Centralized control, when agent based control fails (I = 6)
Evaluation Criteria	Designer-Fabricator Example

Table 3.2: Comparison of the Attributes of Negotiation - Werkman's Model

The system maintains a history of the agents' dialog as the proposal and negotiation process proceeds. This history is later used by the agents for negotiation purposes. The user uses this history of the agents' dialog along with the recommendation of the agents proposals do decide the best choice of connection. This history is also used by the arbitrator to generate arguments during the negotiation process.

Werkman has demonstrated the negotiation process using the prototype Designer,

Fabricator, Interpreter (DFI) system to choose beam-to-column connections acceptable by the structural designer agent, the fabricator agent and the erector agent.

3.3.3 Lander's Negotiation Architecture

Comparison of Lander's Negotiation Architecture with the attributes of negotiation are presented in Table 3.3. Each of these attributes are briefly discussed below.

Lander provides a cooperating experts framework (CEF) designed to support the cooperating experts problem solving (CEPS) paradigm, by providing communication and conflict resolution structures and protocols for cooperative interaction in her computational model. Her framework provides a very flexible conflict resolution format compared to Sycara's model, providing multiple resolution strategies, using the characteristics of the conflict situation to choose one.

The CEF framework only analyze the symptoms of the conflict. The mapping of symptoms to strategies is done by the agents involved in a conflict rather than by a centralized controller. She argues that the cooperating experts' problem-solving paradigm is extremely important in real-world situations, because they bring diverse sources of knowledge. If agents are built that can work together even when the individual agents do not fully understand the entire task, we can begin to look at problems with a whole new level of task complexity.

The CEPS paradigm allows for modularity in the sense that each agent can be implemented, debugged, tested and maintained independently. Agents can be added and deleted with minimal effort. Different sets of agents can be combined to handle different problems or customized to a particular situation. The CEPS agents can also be dynamically arranged to fit the problem solving situation.

In contrast to other systems, Lander treats negotiation as a schedulable activity within a larger problem-solving context. Her argument is that in cases where the importance of quickly finding an acceptable solution outweighs the expense of halting other operations it is not unreasonable to focus on negotiation. But, in the general case there is no reason for all activity to come to a halt because of a conflict. The conflict resolution task is performed asynchronously by the system.

The agents are directly involved in the conflict resolution process and do not expect

Attributes	**LANDER'S Model**
Computational Model	Based on Multiple Resolution Strategies using characteristics of conflict situation to choose one
Central Monitor	Provides control framework
Conflict Detection	Central framework detects types of conflicts
Conflict Notification	Central framework notifies agents about conflicts
Conflict Resolution	Done through mapping conflict symptoms to strategies by agents
Use of Design Rationale	Enables revision of solutions during conflict situation
Negotiation Mechanism	Compromise Negotiation or Integrative Negotiation
Negotiation Strategy	Agents are directly involved in the conflict resolution process without central assistance
Evaluation Criteria	Steam Condenser Example (Partially Implemented)

Table 3.3: Comparison of the Attributes of Negotiation - Lander's Model

any central guidance. Agents keep local histories of their actions and intermediate processing results to enable the revision of solutions in response to conflict situations. Solutions generated by the CEF agents must be approved by all interested agents. There are often difficult tradeoffs that must be made in order to satisfy the various perspectives of different agents.

Lander has used the CEF as the basis for a system that does parametric design of steam condensers, STEAMER [Lander et al 1991b]. STEAMER is partially implemented in the CEF framework. She has identified six agents: pump, heat exchanger, motor, platform, shaft and v-belt, and condenser. The agents have been partially implemented.

3.3.4 Klein's Negotiation Architecture

Comparison of Klein's Negotiation Architecture with the attributes of negotiation are presented in Table 3.4. Each of these attributes are briefly discussed below.

In his computational model, Klein explicitly separates the conflict resolution expertise from the domain-level design expertise. In his model, agents in a cooperative design system can be viewed as being made up of a design component that can update and critique designs, and a conflict resolution component that resolves design agents' conflicts. The conflict resolution component is identical for all design agents. The domain-level design component is distinct for each agent. The conflict resolution expertise is organized into an abstraction hierarchy of conflict types and conflict resolution strategies. A central controller is used to determine the type of conflict and map it to its appropriate strategy.

The central controller detects and notifies conflicts. It also plays a role in mapping the conflict type to the appropriate conflict class from the conflict hierarchy and executing the pieces of advice associated with this class. The conflict resolution component of the agent that first detects a given conflict is given the lead in the conflict resolution process. Klein's main argument is that in his architecture he has given first class status to the conflict resolution expertise. By first class status, he means that he has explicitly separated the conflict resolution expertise from the domain expertise, which other systems fail to do.

Attributes	**KLEIN'S Model**
Computational Model	Based on hierarchy of conflict types and conflict resolution strategies
Central Monitor	Determines conflict types & maps appropriate strategy
Conflict Detection	Done by Central Monitor
Conflict Notification	Done by Central Monitor
Conflict Resolution	Done by instantiation of conflict classes in the conflict hierarchy
Use of Design Rationale	Used to support conflict resolution, avoids backtracking and secondary conflicts
Negotiation Mechanism	Conflict resolution strategy associated with that conflict class is tried
Negotiation Strategy	Centralized control
Evaluation Criteria	LAN Design Example

Table 3.4: Comparison of the Attributes of Negotiation - Klein's Model

Klein advocates the use of design rationale to support the conflict resolution process. He says that the use of design rationale is useful during the conflict resolution process and helps avoid backtracking and secondary conflicts. His main negotiation mechanism is to try to match the conflict situation to the appropriate conflict class in the conflict hierarchy and execute the pieces of advice associated with that class. When a conflict occurs he finds the most specific conflict class that subsumes that conflict, and tries the conflict resolution strategies associated with that class. If none of these strategies are successful, the more general but less efficient strategies associated with that class are tried by moving up the conflict hierarchy. The negotiation strategy has a centralized monitor which controls the negotiation process.

Klein has implemented a system that currently creates designs for Local Area Networks (LAN). It has six agents, and these agents are implemented as expert systems.

3.3.5 Kannapan's Negotiation Architecture

Comparison of Kannapan's Negotiation Architecture with the attributes of negotiation are presented in Table 3.5. Each of these attributes are briefly discussed below.

Kannapan's computational model is based on utility functions and economic negotiation. His model is different from the others, because there is no iteration involved during the negotiation process. All the other approaches have some means of proposal generation, justification, argumentation and proposal modification phases that each agent undertakes during the negotiation process. In Kannapan's architecture, the negotiation strategy is "one shot". Once there is a conflict, the parameter which is responsible for the conflict is found and its utility value computed. Also, how this conflict parameter affects the decision parameter is identified. The utility value of the conflict parameter is then propagated so that a suitable utility value for the decision parameter could be computed, thereby resolving the conflict.

There is no central monitor present and conflicts are detected by the parameter dependencies. There is no conflict notification mechanism present. Conflict resolution is done in two phases. First, the conflict parameters are identified and also

their dependency relationships with the decision parameters. Then, the utility values are propagated from the conflict parameters to the decisions parameters using this relationship. Hence, the conflict resolution process is non-iterative.

Attributes	**KANNAPAN'S Model**
Computational Model	Based on Utility Functions & Nash/Kalai-Smorodinsky solutions
Central Monitor	NONE
Conflict Detection	Done through parameter dependencies
Conflict Notification	NONE
Conflict Resolution	Done by propagation of Utility values through dependency relationships
Use of Design Rationale	None
Negotiation Mechanism	Non-Iterative and done by axiomatically derived agreement points
Negotiation Strategy	One shot without Iteration
Evaluation Criteria	Poppet Relief Valve Example

Table 3.5: Comparison of the Attributes of Negotiation - Kannapan's Model

Kannapan has used the design of a poppet relief value to demonstrate the negotiation process and his computational model.

3.4 Summary

This chapter compared the negotiation architecture of some selected negotiation systems. Through this study we are able to better understand the computational model, the conflict detection, notification and resolution mechanisms as well as the negotiation strategies used in these various systems. We are also able to understand

how the various negotiation systems are evaluated. In the next chapter we will compare the communication architecture of these negotiation systems.

Chapter 4

A Comparative Analysis of Communication Architecture

4.1 Overview

This chapter presents the comparative analysis of negotiation systems, specifically the communication architecture. Section 4.2, describes the main attributes of communication used in the negotiation process. In Section 4.3, these attributes are used to compare the selected negotiation systems [Sycara 1991], [Werkman 1990], [Lander et al 1991b], [Klein 1991], and [Kannapan & Marshek 1991].

4.2 Attributes of Communication

4.2.1 Communication Medium

One of the major attributes of communication is the type of communication medium used. The two major types of communication medium used are shared memory and message passing. In the shared memory type of communication medium all the agents share a central shared memory for communication. In the message passing type of communication medium, the agents pass messages back and forth

for communication. One of the most widely used shared memory approaches is the blackboard model [Nii 1986a], [Nii 1986b].

4.2.2 Communication Protocol

The other major attribute of communication, is the kind of communication protocol being used. The communication protocol is necessary for an effective means of communication between the agents. One of the types of communication protocol being followed by researchers is the one based on Speech Acts [Austin 1962], [Searle 1969].

The communication protocol was derived from a series of speech related social actions that occur between agents. Each social interaction that an agent might enter into contains a case structure (to, from), preconditions (necessary agent conditions) and postconditions (result of successfully performing the action). Thus, communication in terms of globally acceptable social actions provide for a plan-based approach for communication. This means that once an agent receives a specific message from a sending agent, the receiving agent will know what type and form of response message with which it is expected to reply. This makes for short explicit messages and reduces extraneous message overhead.

4.2.3 Communication Language

Another attribute of communication is the kind of communication language being used. Generally, the communication language used in any system is based on the communication protocol. There can be two different types of communication languages. The first type of communication language is used between agents. The other type of communication language is used between the central monitor and the agents for their communication. In some systems there could be only one type of shared communication language which is used for both the communication between agents as well between the central monitor and the agents.

4.3 Comparison of Communication Architecture

4.3.1 Sycara's Communication Architecture

An analysis of Sycara's Communication Architecture using the attributes of communication is presented in Table 4.1. Each of these attributes are briefly discussed below.

Attributes	SYCARA'S Model
Communication Protocol	Under development
Communication Medium	Under development
Communication Language	Under development

Table 4.1: Comparison of the Attributes of Communication - Sycara's Model

Sycara has not done much work regarding the communication architecture that is necessary to support negotiation, at least, for example, when compared to Werkman [Werkman 1992] or Lander [Lander et al 1991b]. Her main focus is on the negotiation process and has mentioned that work in the communication area is still under development. In [Sycara 1991] she presents the basic communication protocol between two agents which is briefly explained in Section 2.2.8.

4.3.2 Werkman's Communication Architecture

An analysis of Werkman's Communication Architecture using the attributes of communication is presented in Table 4.2. Each of these attributes are briefly discussed below.

Werkman's communication architecture clearly specifies the communication protocol, the communication medium and the communication language he has used. In his system, agents communicate by means of a centralized communication medium called the blackboard [Nii 1986a], [Nii 1986b]. The blackboard scheme allows agents to post messages as well as read messages from other agents. The message structure

consist of: to whom the message is sent, from whom the message is received and the message content. The use of such a scheme allows the system to maintain a history of agents' dialog as the proposal and negotiation process proceeds.

Attributes	**WERKMAN'S Model**
Communication Protocol	Based on Speech Acts
Communication Medium	Black Boards
Communication Language	Message Primitives

Table 4.2: Comparison of the Attributes of Communication - Werkman's Model

Agents use a common language for interagent communication that consists of primitive messages. This common language allows for an effective form of negotiation between agents which allows them to reason about beliefs of other agents. This interagent language must allow for the expression of agent intentions at some level of abstraction. To achieve this, the interagent language was based on a communication protocol derived from speech act theory [Austin 1962], [Searle 1969].

4.3.3 Lander's Communication Architecture

An analysis of Lander's Communication Architecture using the attributes of communication is presented in Table 4.3. Each of these attributes are briefly discussed below.

Lander's Cooperating Experts Framework (CEF) has a distinct communication protocol, communication medium and communication language. The CEF framework uses blackboards as a global communication medium. It provides a set of blackboards which can be accessed freely by all agents. There are three different kinds of blackboards supported by the CEF framework, namely the shared databases blackboard, the proposal blackboard and the negotiations blackboard. The shared databases blackboard is a place where static public information such as component catalogs and global constraints are stored. The proposals blackboard is a place where

dynamic public information such as partial or complete solutions are stored. Comparisons of proposals and other aspects of negotiation are stored in the negotiations blackboard.

Attributes	**LANDER'S Model**
Communication Protocol	Asynchronous Communication
Communication Medium	Shared databases BB Proposals BB Negotiations BB
Communication Language	Based on Object Definition Language

Table 4.3: Comparison of the Attributes of Communication - Lander's Model

All communication in the CEF framework takes place asynchronously through the blackboards. In the CEF framework, the negotiation process can be scheduled as a separate activity and the entire system need not wait when this activity takes place.

The information stored on the blackboards is generally represented in a common language shared by all agents. If this language is different from the internal language used by the agents then translation procedures must be provided to share information among agents. The CEF framework uses a generic, hierarchical, structured Object Definition Language [Lander & Lesser 1989] provided by the GBB blackboard development system.

4.3.4 Klein's Communication Architecture

An analysis of Klein's Communication Architecture using the attributes of communication is presented in Table 4.4. Each of these attributes are briefly discussed below.

Klein's communication architecture supports only a communication medium and communication language, but does not support any communication protocol. This is because, the communication language is very simple and is not based on any explicit

communication protocol. The agents use a central blackboard [Nii 1986a], [Nii 1986b] to refine and critique abstract component descriptions.

Attributes	**KLEIN'S Model**
Communication Protocol	NONE
Communication Medium	Black Boards
Communication Language	Action & Query Language

Table 4.4: Comparison of the Attributes of Communication - Klein's Model

In Klein's system there are two kinds of languages used. The first language, called the query language, is used by the CR component to query the design agents in order to map the conflict type to the appropriate conflict class hierarchy. The other language used in his system is the action language that is used to execute the CR plans.

4.3.5 Kannapan's Communication Architecture

An analysis of Kannapan's Communication Architecture using the attributes of communication is presented in Table 4.5. Each of these attributes are briefly discussed below.

Kannapan has a very simple communication architecture for his negotiation system, because his negotiation is done by propagating utility values and is not iterative as in the other systems.

The communication protocol used in Kannapan's system is based on a combination of forward and backward propagation of utility values. There is no explicit communication medium. The communication language passes messages containing utility values back and forth, and is not very explicit.

Attributes	**KANNAPAN'S Model**
Communication Protocol	Based on forward and backward propagation of utility values
Communication Medium	No explicit medium for communication
Communication Language	Implicit messages conveying utility values

Table 4.5: Comparison of the Attributes of Communication - Kannapan's Model

4.4 Summary

This chapter compared the communication architecture of some of the negotiation systems. Through this study we are able to better understand the communication protocol, communication medium and communication language used in these various systems. In the next chapter we will compare the agent architecture of some of these negotiation systems.

Chapter 5

A Comparative Analysis of Agent Architecture

5.1 Overview

This chapter presents the comparative analysis of negotiation systems, specifically the agent architecture. Section 5.2 describes the main attributes of the agents that participate in the negotiation process. In Section 5.3, these attributes are used to compare the selected negotiation systems [Sycara 1991], [Werkman 1990], [Lander et al 1991b], [Klein 1991], and [Kannapan & Marshek 1991].

5.2 Attributes of Agents

5.2.1 Agent Type

Agent type is one of the major attributes of the agents. There are three different types of agents. The first type of agent is the totally independent, autonomous agent. This type of agent does not depend on the central controller for scheduling, negotiation etc. These agents have all the knowledge necessary to detect conflicts, to resolve conflicts (negotiate) and for scheduling. This type of agent is shown in column

8 of Table 5.1. Some agents in this category, column 7 of Table 5.1, only have the capability for scheduling and conflict resolution but expect to be notified of conflicts.

Agent Can	1	2	3	4	5	6	7	8
Detect Conflicts	N	Y	N	Y	N	Y	N	Y
Resolve Conflicts	N	N	Y	Y	N	N	Y	Y
Schedule	N	N	N	N	Y	Y	Y	Y

Table 5.1: Generic Agent Types

The second type of agent is the semi-autonomous agent. This type of agent depends, to a certain extent, on the central controller. They have the knowledge to detect conflicts, and to resolve conflicts, but do not have any scheduling capabilities. This type of agent is shown in column 4 of Table 5.1. Some agents in this category, column 3 of Table 5.1, only have the capability for conflict resolution but expect to be notified of conflicts and also scheduled to execute.

The third type of agent is the totally dependent type of agent. This type of agent depends entirely on the central controller. They do not have any knowledge to detect conflicts, or to resolve conflicts or for scheduling. This type of agent is shown in column 1 of Table 5.1.

Agents in Columns 2, 5, and 6 of Table 5.1 are not generally considered because the minimum functionality they should perform is to resolve conflicts.

5.2.2 Agent Grainsize

This attribute characterizes the grainsize of the agents, where by "grainsize" we mean the complexity, knowledge content, and the general capabilities of the agents. This attribute is dependent on the agent's type. If the agent is a fully autonomous type of agent, then the agent's grainsize will be fairly large (complex agents). If the agent is a semi-autonomous type of agent, then the agent's grainsize will be medium (not very complex agents). If the agent is a totally dependent type of agent, then the agent's grainsize will be small (simple agents).

5.2.3 Agent's Domain Knowledge

Another attribute of the agents is whether the agent has explicit domain knowledge defined. If the domain knowledge is explicitly defined, then we are concerned with what kind of a scheme is used to represent it, e.g., rules, frames, etc. There are cases, where the domain knowledge, control knowledge and negotiation knowledge are not explicitly defined.

5.2.4 Agent's Negotiation Knowledge

Another attribute of the agents is whether the agent has explicit negotiation knowledge defined. If the negotiation knowledge is explicitly defined, then we are concerned with what kind of a scheme is used to represent it, e.g., rules, frames, etc.

5.2.5 Agent Privacy

This attribute characterizes how much each agent knows about the other agents. Some of the information that an agent might know about another agent includes, the other agent's functionality, e.g., advice, criticism, estimation, etc.; the other agent's domain, e.g., material, process, inspection, etc.; and the other agent's point of view, e.g., cost, strength, safety, etc. Accordingly, agent privacy can be classified into three major categories.

The first category contains agents, that do not have any knowledge about other agents. The second category contains agents that have partial knowledge about the other agents, e.g., other agent names etc. The third category contains agents that have complete knowledge about the other agents. The agents in this category have complete knowledge of the other agent's goals, belief structure and constraints.

5.2.6 Agent Examples

This attribute characterizes the different kinds of agents each researcher has implemented. The different kinds of agents are based on three aspects. The first aspect is the agent's functionality, that is, what kind of a task it can perform. The

second aspect is the agent's target domain, that is, what kind of a domain it is working in. The third aspect is the agent's point of view.

For example, in the design of an aircraft [Sycara 1991], the different kinds of agents involved are a aerodynamic agent, a structural engineering agent, a manufacturing agent, a marketing agent, etc. In the design of a steam condenser [Lander et al 1991b], some of the different kinds of agents involved are a motor agent, a pump agent, a heat exchanger agent, a condenser agent, a platform agent etc.

5.3 Comparison of Agent Architecture

5.3.1 Sycara's Agent Architecture

An attribute-based analysis of Sycara's Agent Architecture is presented in Table 5.2. Each of these attributes are briefly discussed below.

In Sycara's negotiation system, the agents are independent and autonomous. The agents do not need the assistance of a central monitor for negotiating. They communicate directly with each other to resolve their conflicts. Due to this independent nature, the grainsize of the agents is fairly large. The agents in her system have the capability to represent and maintain knowledge about other agents' goals/beliefs, reason about other agents' goals/beliefs, and influence other agents' beliefs by exchanging missing information, justifications and arguments. Agents have a belief structure that represents the importance and relationship among goals. The belief structure also includes attributes of importance, feasibility, and contribution. There is also a representation of the preference structure of an agent which expresses the utility for each attribute in the goal graph (belief structure).

Generally, agents have both domain knowledge and negotiation knowledge embedded in them. In Sycara's system, the agents' domain knowledge is represented along with the agents' negotiation knowledge. Negotiation knowledge comes from diverse sources such as cased based reasoning (CBR), multi-attribute utilities, and constraint relaxation.

Sycara's agents have knowledge about other agents' goals/beliefs, and they can reason about other agents' goals/beliefs as well as influence other agents beliefs by

Attributes	**SYCARA'S Model**
Agent Type	Independent, Autonomous
Agent Grainsize	Large
Agent Domain Knowledge	Represented explicitly
Agent Negotiation Knowledge	Represented explicitly using CBR, Utilities, Constraints
Agent Privacy	Have knowledge about other agents
Agent Examples	Aerodynamic agent Structural Engineering agent Manufacturing Agent Marketing Agent

Table 5.2: Comparison of the Attributes of Agents - Sycara's Model

exchanging missing information, justification and arguments. Hence, in her system, agents do not have much privacy because all agents have knowledge about other agents in the system. In the explanation of her approach, using the example of designing a turbine blade, the important agents involved include, the aerodynamics agent, the structural engineering agent, the manufacturing agent and the marketing agent.

5.3.2 Werkman's Agent Architecture

An attribute-based analysis of Werkman's Agent Architecture is presented in Table 5.3. Each of these attributes are briefly discussed below.

Werkman in his Designer, Fabricator, Interpreter (DFI) System has modeled the agents to be semi-autonomous. In his system, once the connection to be evaluated is selected, each agent evaluates the connection from its own specific viewpoint. During this evaluation process, agents can propose alternate connections for consideration which may require negotiation among the agents. Werkman's agents do not have much knowledge about other agents in the system. This makes them look small

compared to Sycara's agents.

Attributes	**WERKMAN'S Model**
Agent Type	Semi-Autonomous
Agent Grainsize	Medium
Agent Domain Knowledge	Represented explicitly
Agent Negotiation Knowledge	Represented explicitly
Agent Privacy	No knowledge about other agents
Agent Examples	Designer Agent Fabricator Agent Erector Agent Arbitrator Agent

Table 5.3: Comparison of the Attributes of Agents - Werkman's Model

Werkman's agents are implemented as knowledge based systems. Each agent's domain and negotiation knowledge is represented explicitly. The system evaluates and suggests alternative connections based on multiple agent viewpoints, e.g., design, fabrication, erection etc. Each agent viewpoint is further decomposed into several unique agent issues. The issues are based on different aspects of connections such as functional aspects, component aspects, fastener aspects, etc. The importance of an agent issue depends on which agent viewpoint one takes (designer, fabricator, or erector) within the context of a specific connection evaluation.

With respect to privacy, the agents do not have much knowledge about other agents. During the negotiation process, the arbitrator agent supplies whatever missing knowledge is required by the agents involved in the negotiation process. Werkman has implemented his negotiation system, the DFI system, to address the lack of interaction among structural designers, fabricators and erectors in dealing with beam-to-column connections. The agents involved in his system are the designer agent, fabricator

agent, erector agent and the arbitrator agent.

5.3.3 Lander's Agent Architecture

An attribute-based analysis of Lander's Agent Architecture is presented in Table 5.4. Each of these attributes are briefly discussed below.

In Lander's Cooperating Experts Framework (CEF), the agents are fully functional knowledge-based systems which can solve problems in their limited domain independently. The agents do their own internal scheduling and have private data, knowledge, goals and history mechanisms. Due to these capabilities, the agents are highly independent and autonomous.

Although the agents can operate independently as a separate problem-solving entity, they have certain specific capabilities which allow them to act as a member of a team as well. Some of these capabilities include a shared communication language; internal knowledge representation, which captures sufficient goal and history information to allow for cooperative solution revision; provision for sharing information; mechanism for incorporating externally produced partial solutions; and mechanism for negotiation the settlement of conflicts. Due to these considerations the agents are fairly large knowledge based systems.

Lander's agents have the capability to explicitly represent their evaluation criteria, constraints and goals. This is needed so that this information can be communicated to other agents as part of the conflict resolution process. The agents also have their domain knowledge explicitly represented. Agents keep local histories of their actions and intermediate processing results to enable the revision of solutions in response to conflict situations. Local processing results are not accessible to other agents unless they are explicitly shared. The agents contain a relatively large amount of knowledge about other agents.

Lander has used the CEF framework as the basis for a system that does parametric design of steam condensers, STEAMER. The various agents involved in this design process include a motor agent, a pump agent, a heat exchanger agent, a condenser agent, and a platform agent.

Attributes	**LANDER'S Model**
Agent Type	Independent, Autonomous
Agent Grainsize	Large
Agent Domain Knowledge	Explicitly represented using Knowledge Base Systems
Agent Negotiation Knowledge	No explicit representation
Agent Privacy	Complete knowledge about other agents
Agent Examples	Motor Agent Pump Agent Heat Exchanger Agent Condenser Agent Platform Agent

Table 5.4: Comparison of the Attributes of Agents - Lander's Model

5.3.4 Klein's Agent Architecture

An attribute-based analysis of Klein's Agent Architecture is presented in Table 5.5. Each of these attributes are briefly discussed below.

In Klein's negotiation system, the agents are semi-autonomous. The presence of a central controller alleviates the need for the agents to be independent and autonomous. Klein [1991] has developed a hierarchy of conflict types and conflict resolution strategies. This conflict resolution knowledge is explicitly represented. The agents domain-level expertise is separately represented from the conflict resolution (negotiation) expertise. Only in Klein's agents is this separation very explicit. He argues that to represent and reason about conflict resolution expertise separately gives it first-class status, which, so far, has been given only to domain knowledge.

Giving CR expertise distinct first-class status enables the conflict resolution component to succinctly capture the most useful, general, conflict resolution principles. It also allow bodies of domain-level design expertise to be represented in "pure" form without having to anticipate potential conflicts with each other. This separation of conflict resolution and domain knowledge is claimed to increase the flexibility and generality of multiple-expertise knowledge-based systems. Whenever there is a conflict, the central controller determines the type of conflict and maps it to its appropriate strategy.

Klein's agents do not have complete knowledge about the other agents in the system. They only have partial knowledge about the other agents. Klein has implemented a system called the cooperative design engine (CDE), which creates designs for Local Area Networks. It consists of a set of design agents. Agents can either participate in deciding (e.g., advice, selection, estimation) a particular design parameter, or criticizing an existing design from a particular design perspective. Design agents cooperate by refining and critiquing abstract component descriptions stored on a central blackboard.

Design agents are implemented as rule-based expert systems. In the current system there are six agents which have knowledge about available LAN technology, security, reliability, vendor needs, expandability and economics. The available LAN technology agent knows about existing LAN technologies and how to combine them

Attributes	**KLEIN'S Model**
Agent Type	Semi-Autonomous
Agent Grainsize	Small
Agent Domain Knowledge	Explicitly represented using Expert Systems
Agent Negotiation (CR) Knowledge	Explicitly Represented
Agent Privacy	Partial knowledge about other agents
Agent Examples	Local Area Network Agent Security Agent Vendor Needs Agent Expandability Agent Reliability Agent Economics Agent

Table 5.5: Comparison of the Attributes of Agents - Klein's Model

into working system given detailed specifications. The remainder of the agents offer constraints on the specifications and critique the emerging design from their particular perspectives. The CR component in each agent includes a conflict class taxonomy with a total of 115 conflict classes.

5.3.5 Kannapan's Agent Architecture

An attribute-based analysis of Kannapan's Agent Architecture is presented in Table 5.6. Each of these attributes are briefly discussed below.

In Kannapan's concurrent engineering schema [Kannapan & Marshek 1991], design agents are allowed to make decisions independently. The agents involved here are semi-autonomous and not very big compared to Sycara's or Lander's agents. Each agent has utility functions associated with its conflict parameters, which can be obtained from many sources. Some of these sources include direct questioning of the decision maker, derivation from specific design knowledge, experimentation and engineering principles, or retrieval from previous similar cases.

Attributes	**KANNAPAN'S Model**
Agent Type	Semi-Autonomous
Agent Grainsize	Medium
Agent Domain Knowledge	Explicitly represented using Utilities
Agent Negotiation Knowledge	Explicitly Represented using Utilities
Agent Privacy	No Knowledge about other agents
Agent Examples	Valve-flow agent Valve-cracking agent Pipe-enclosure agent Helical-spring agent

Table 5.6: Comparison of the Attributes of Agents - Kannapan's Model

Kannapan's agents do not have any knowledge about other agents and have only knowledge about the utility values associated with its own conflict parameters. Kannapan has demonstrated his ideas by building a concurrent engineering system for the design of the poppet relief valve. The various design agents involved in this system are the valve-flow design agent, the valve-cracking design agent, the pipe-enclosure design agent, and the helical-spring design agent.

5.4 Summary

This chapter compared the agent architecture of some of the negotiation systems. Through this study we are able to better understand the agent types, agent grainsize, agents domain knowledge representation, agents negotiation knowledge representation, agents privacy, and the examples of agents used in these various systems. In the next chapter, we will look at the domain of our negotiation system, the concurrent engineering I3D system [Victor et al, 1993].

Chapter 6

Domain of Demonstration: I3D System

6.1 Overview

In Section 6.2, a brief introduction is given to the the domain of the demonstration system, the I3D System, and the context of its development. In Section 6.3, the I3D System's architecture is explained. In Section 6.4, the different expert systems developed to support the I3D System, and their functionality is discussed. In Section 6.6, the limitations of the I3D system and the possible improvements are addressed.

6.2 Introduction

The I3D System, *Intelligent, Integrated, Interactive Design System*, was developed at the Center for Intelligent Processing of Materials, Worcester Polytechnic Institute. This system was developed as an integrated solution for designers. The system concurrently provides product expertise along with the CAD design and engineering (Finite Element) analysis. The expertise provided includes areas such as manufacturing, inspection, cost, reliability and durability. This development work was done for the U.S. Army Materials Technology Laboratory located at Watertown,

Massachusetts. The I3D system development team consisted of faculty and students from Computer Science, Manufacturing, Materials, and Electrical Engineering. Development took place over a period of about four months. This work has also been described elsewhere [Zenger et al 1993], [Victor et al, 1993].

The system interacts with a designer sitting at a workstation as shown in Figure 6.1. As the designer moves through requirements specification, conceptual design and detailed design of the part to be made from powder ceramic material, the system graphically displays the state of the design on the screen. It makes appropriate assumptions about design decisions not yet made in order to be able to continuously display the component during the various stages of design.

As requirements are given and design decisions are made, the system provides feedback about the design from several different points of view. Intelligent agents, expert systems, display these comments on the screen. Each agent is given a chance to respond. Comments might include estimation of cost, advice about which ceramic powder to use, and information about the Inspection process required for the part.

The system is intended to be extensible, so every effort was made to allow different agents to be added easily. We have added intelligent agents to test this, as well as adding an agent which performed a simulation to determine the effects of compaction and sintering on part size, density and cracking. Additions were found to be easy.

The system was mainly intended to be an investigation of what knowledge, what types of agents, and what type of control strategy would be required for an interactive design system of this kind for powder ceramics. It was not intended as a "complete" concurrent engineering system. Consequently we concentrated on many aspects of the design process, but for a limited class of parts.

A unique feature of the I3D system is that it can provide considerable feedback about a variety of "downstream" aspects during the conceptual design phase. For example, one agent performs cost estimation during the conceptual design phase, while another estimates cost during the detailed design phase. Although the cost during the conceptual phase is approximate, it does provide useful information, allowing alternative designs to be compared.

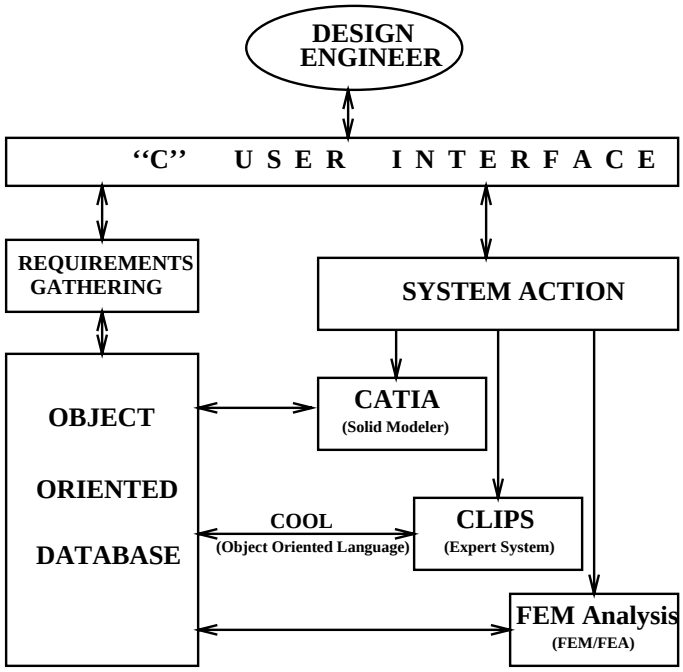

Figure 6.1: I3D System Architecture

6.3 System Architecture

The I3D system is organized in such a way that the data needed during the whole design process is retrieved from a central area. The requirements and other system parameters are stored in an Object Oriented Representation using CLIPS Object Oriented Language (COOL) [Giarratano 1991a], [Giarratano 1991b]. The various expert systems also access these objects for their processing. The logical data flow of the system is shown in Figure 6.2.

In order to prevent conflict between the intelligent agents and to remove the need for negotiation, three techniques were used. The first was to impose a strict control regime, such that all agents were "fired" as a group (i.e., after the user's request for analysis) in a predetermined sequence, with each agent coming after those on which it depends. In Figure 6.2, this predetermined sequence is shown in the clock-wise fashion. The second technique was to ensure that each agent that made a decision was the only one responsible for it, and that it made its decision with all relevant

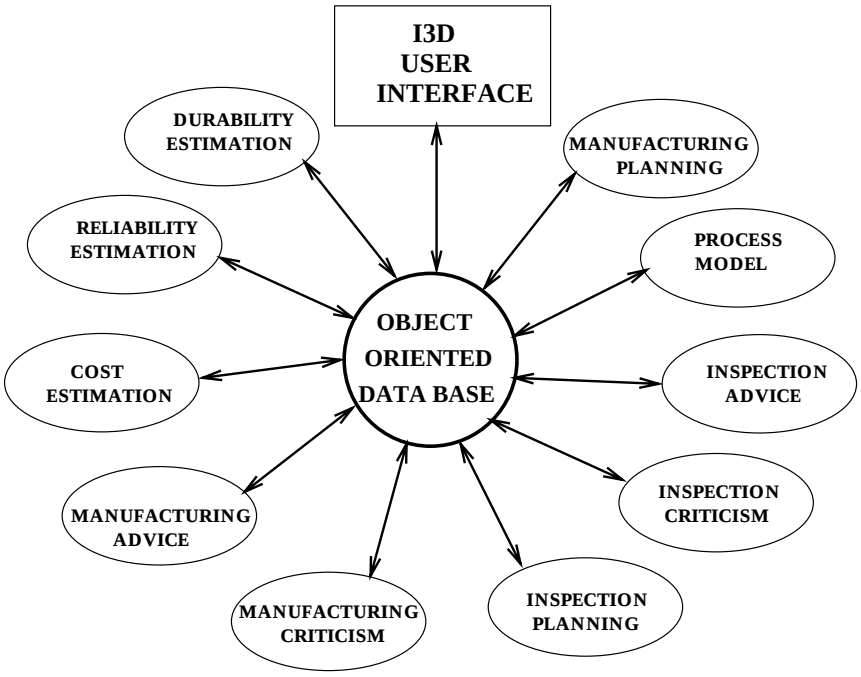

Figure 6.2: Organization of Expert Systems and other Sub-Systems

goals (e.g., cost, strength) being considered at that time. The third was to allow the user to override decisions reached by agents, i.e., the user was in control of resolving any conflicts between the agents.

6.4 Expert Agents

An extremely important aspect of this system is the choice of roles that the intelligent agents can play [Brown 1992]. The roles used were to provide Advice, Criticism, Planning, Selection and Estimation.

• An Advisor provides information about what to do. For example, it suggests a portion of the design, such as a value for a length.

• A Critic comments on possible problems with existing design decisions. For example, it points out a non-standard chamfer angle.

• A Planner produces a choice of actions and their sequencing. For example, it can determine sensor placement for inspection or a processing sequence.

• A Selector picks one item from a list. For example, it could select a material or a particular process.

• An Estimator can estimate derived values. For example, it can estimate cost or reliability.

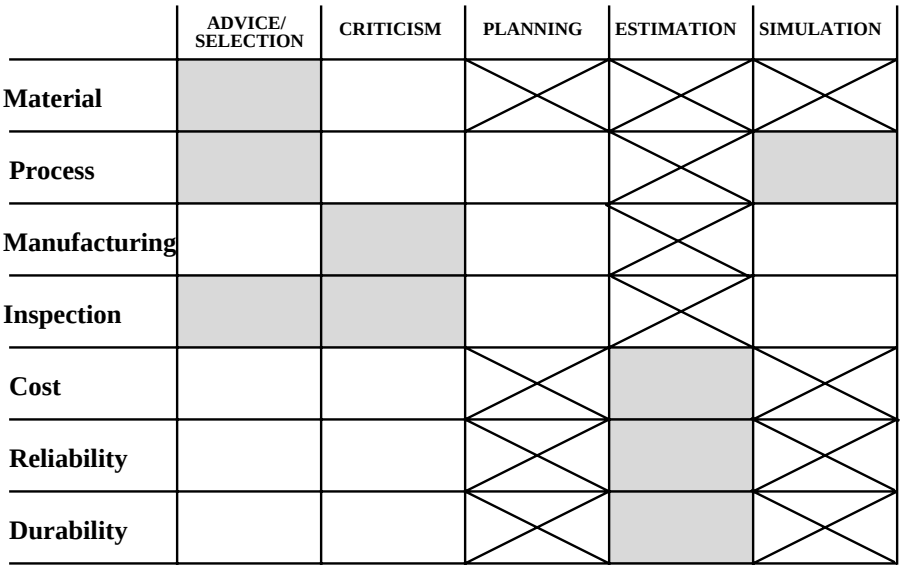

Figure 6.3: Roles/Aspects of Expert Agents (Conceptual Phase)

These roles for intelligent agents, can be contrasted with the "topic" that they make comments about. We refer to these as Aspects, and they include Material, Process, Manufacturing, Inspection, Cost, Reliability and Durability [Brown et al 1993a]. The roles and aspects define a matrix of possible agents as shown in Figure 6.3.

In addition, we can consider Simulation as another role, even though it is not provided by an intelligent agent. In I3D there are 16 intelligent agents in total, with 8 assigned to the conceptual design (Figure 6.3) and 8 to the detailed design (Figure 6.4). Some examples are given below.

6.4.1 Expert Systems for Advice and Selection

There is an agent to give advice regarding the choice of the inspection method. Another selects the best materials suitable for the component being designed. A third selects the best process suitable to manufacture this component. Each system

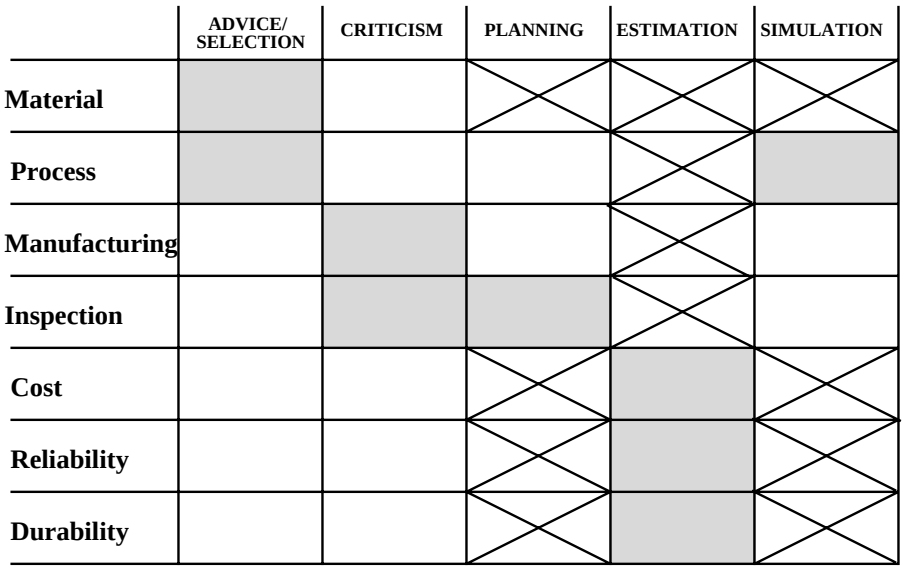

Figure 6.4: Roles/Aspects of Expert Agents (Detailed Phase)

has encoded a great deal of technical information encoded in. The information is encoded in a context dependent way, so that it responds to previous choices and to the requirements on the design.

6.4.2 Expert System for Criticism

There is an expert system for manufacturing criticism and one for inspection criticism. These systems give their criticisms about the manufacturability and inspectability of the component. They point out the potential problems from the point of manufacturing it and also inspecting it.

6.4.3 Expert System for Planning

There are agents for manufacturing and inspection planning. These systems are activated only during the detailed phase of the design process. They plan details of the whole manufacturing process as well as the inspection method for the component.

6.4.4 Expert System for Estimation

There is an expert system for estimating the cost of the component being designed. This system is activated during both the conceptual and detailed phase of the design process. It gives the cost estimate of the materials cost, process cost (primary and secondary), tooling cost, and inspection cost, as shown in Figure 6.5.

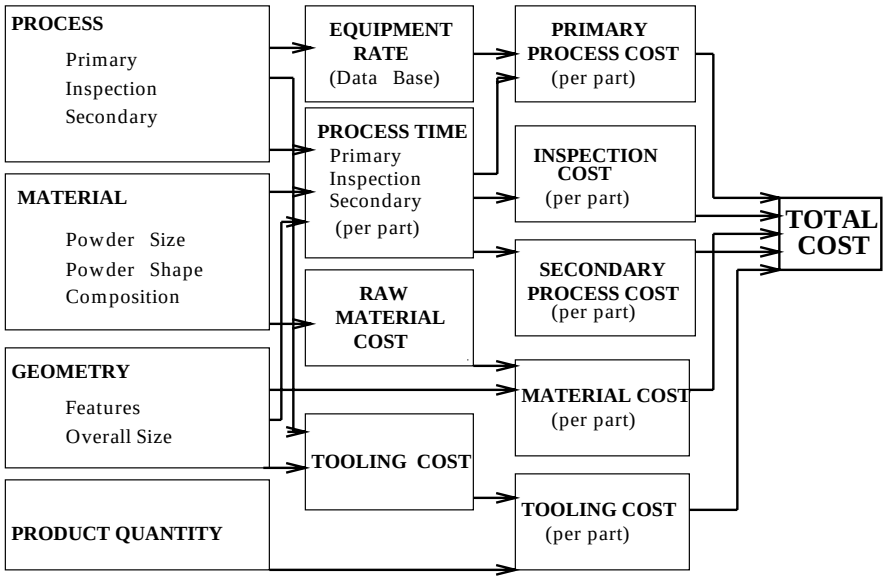

Figure 6.5: Expert System for Cost Estimation

6.5 System Implementation

Building an integrated system according to the general model described above requires use of commercial, open architecture software systems for solid modeling, expert systems development and engineering analysis. This enables easy transfer of the systems developed.

Because the process planning models used for design and control are largely dependent on the solid geometry of the workpart, a CAD driven approach is employed to increase the flexibility of the system for complex geometries.

The solid modeler used for development of I3D is CATIA, a high level solid modeling program from IBM. CATIA is a state-of-the-art system currently in use by many

of the major U.S. aerospace companies. The analysis package used by this system is called CAEDS/IFES (Integrated Finite Element Solver) and is also supplied by IBM.

The expert system building tool chosen was CLIPS (C Language Integrated Production System), a NASA-developed rule-based forward chaining system. This provided an easy interface to programs written in C, as well as providing an object oriented language, the CLIPS Object Oriented Language (COOL), in which we encoded general knowledge about the materials, processes and parts relevant for the design. The state of the design was stored as CLIPS facts, with interagent communication being achieved by CLIPS facts, COOL, and use of files.

The whole system as shown in Figure 6.1, runs under UNIX (AIX 3.1) on an IBM RS/6000 workstation with a color display. The user interface was developed using C and standard UNIX capabilities.

6.6 I3D+: An Extension to the I3D System

6.6.1 Current System: I3D

The current version of the I3D System [Victor et al, 1993], is a working system in the domain of manufacturing and materials engineering. This is a concurrent engineering system where negotiation was completely avoided. This system was chosen to be the domain of our demonstration system, because it is a multi-agent system implemented to automate the real-world manufacturing process of ceramic components.

The I3D System was developed as a concurrent engineering system to primarily automate the manufacturing process of ceramic components. Though the system had a rich potential for conflicts among the various agents, they were avoided during the development phase of the system by the three techniques described in Section 6.3.

6.6.2 Limitations of the I3D System

The I3D System had a couple of limitations. The scheduling strategy was fixed without much flexibility, and it was not easy to change the order in which the agents were executed.

The other issue was that no parameter could be decided by more than one agent. For example, the "material" parameter cannot be decided by two agents with different perspectives. In real-world situations, it is natural for experts to have conflicting views. Avoiding them in the I3D system makes it look a little artificial. Also, it is difficult to add new agents because the system has to be informed about the agents at compile time.

6.6.3 Extensions of I3D: I3D+

These limitations were addressed in the I3D+ System which in addition to being a concurrent engineering system is also a full fledged negotiation system, capable of handling conflicts without user intervention.

The I3D+ system has a high potential for conflicts among the various agents. It allows two agents to decide the value of a single parameter. It also allows one agent to decide a value and another agent to give criticism about this value. The system has a powerful scheduling mechanism based on Agendas. The agenda based scheduling mechanism is very flexible and allows agents to have different scheduling priorities based on their ranking. Hence, by changing the agents' ranking, we can control the order in which the agents are executed. The system has a conflict detection and resolution mechanism built into it. The communication architecture of the system is based on speech act theory [Austin 1962], [Searle 1969]. The system has the capability to explicitly represent the agents' domain knowledge, negotiation knowledge, and redesign knowledge.

Hence, in addition to providing the same functionality as the I3D system, the I3D+ system also has the capability to resolve conflicts. The detailed comparisons between the I3D and I3D+ systems are summarized in Section 7.2.2.

6.7 Summary

This chapter detailed the choice of a domain for the Negotiation Demonstration System, which would meet the requirements of being a multi-agent, concurrent engineering system. It is complex enough to demonstrate the different aspects of

negotiation. The architecture and implementation of the system was discussed along with a brief description of the expert agents involved. Finally, the limitations of the I3D system were discussed, and how we are going to improve these limitations by extending the system to have the capability to negotiate and resolve conflicts. The next chapter presents the architecture and implementation of the I3D+ System.

Chapter 7

I3D+ Negotiation System

7.1 Overview

In Section 7.2, a brief introduction about the I3D+ Negotiation System is given. It will be compared with the original I3D System. In Section 7.3, the I3D+ System architecture is presented. In Section 7.4, the scheduler for the I3D+ System is described and in Section 7.5 the implementation details of the I3D+ System is discussed.

7.2 Introduction

The Concurrent Engineering system, I3D [Victor et al, 1993], described in chapter 6, was extended by introducing conflicts among the various expert agents. The extended system is called the I3D+ System. When the agents in the I3D+ are scheduled to execute, conflicts can arise, and these conflicts are resolved using negotiation. The major claim to this work is that we are using a practical working concurrent engineering system in a real domain. In I3D, negotiation was avoided as all conflicts were avoided. In the I3D+ System, we deal with conflicts; we allow them to exist, and resolve them using negotiation.

7.2.1 Types of Conflicts

Design agents cooperate by updating a shared representation of the design, and by critiquing design commitments made by other design agents. Hence, there are basically two major types of conflicts. The first type occurs when two design agents try to decide the value of a single parameter. The second type occurs when one design agent has negative criticism about the decision made by another design agent. We will see how these types of conflicts are handled in both the I3D as well as the I3D+ System.

7.2.2 Comparison of I3D with I3D+

Comparison of the I3D system with the I3D+ system clearly demonstrates the capabilities of the I3D+ negotiation system. The results of the comparison is shown in Figure 7.1.

FEATURE	I3D	I3D+
Scheduling Strategy	**Fixed Scheduling**	**Agenda based Scheduling Flexible & Dynamic**
Conflicts among agents	**identified & avoided during development**	**High potential for conflicts among agents**
Conflict Detection & Conflict Notification	**NONE**	**YES, Implemented**
Negotiation/CR	**NONE**	**YES, Proposals, Justifications (Based on Sycara's Model)**
Communication between agents	**NONE**	**Direct Communication (Based on Sycara's Model)**
Communication Protocol	**NONE**	**Using Speech Acts (Based on Werkman's Model)**
Explicit Domain Knowledge	**YES**	**YES**
Explicit Negotiation Knowledge	**NONE**	**YES (Based on Klein's Model)**

Figure 7.1: I3D versus I3D+ System

In the I3D system, the potential source of conflicts between agents were identified

and avoided during the development phase. This avoided the need for negotiation and was done using the following three techniques:

• Agents were executed in a fixed predetermined sequence. Each agent came after those on which it depended

• Each agent decided the value of a single parameter. Other agents only used this value for their processing.

• Allowing the user to override decisions reached by agents, i.e., the user was in control of resolving any conflicts between the agents.

In the I3D+ system, the natural sources of conflicts are preserved. Agents are allowed to execute, without any predetermined sequence, by an agenda based scheduler. More than one agent is capable of deciding the value of a particular parameter. There are agents which could give negative criticism about another agent's decision. Hence, the I3D+ system had a high potential for conflicts among agents.

There was no explicit conflict detection mechanism present in the I3D system because there were no conflicts. In the I3D+ system, there exists explicit conflict detection and notification mechanisms to detect and notify conflict situations, respectively.

In the I3D system, there was no conflict resolution or negotiation mechanism present as there were no conflicts to resolve. In the I3D+ system, there exists a conflict resolution mechanism, which resolves conflicts by passing intentions and justifications between agents. This exchange of intentions and justifications between agents is based on Sycara's Model as explained in Section 2.2.

In the I3D system, there was no direct communication between agents. The agents only communicated through the shared database. In the I3D+ system, there is direct communication between agents. This kind of direct communication between agents is based on Sycara's Model as explained in Section 2.2.

In the I3D+ system, agents communicated using a communication protocol derived from Speech Acts Theory [Austin 1962], [Searle 1969]. This communication protocol is based on Werkman's Model, explained in Section 2.5, who also used Speech Acts.

In the I3D system, the agents had explicit domain knowledge only and they did not have any explicit negotiation knowledge. In the I3D+ system, the agents have

the capability to represent both the domain knowledge, as well as the negotiation knowledge. This explicit separation of domain knowledge and negotiation knowledge was advocated by Klein in order to give first class status to negotiation.

Overall, the I3D+ system, in addition to offering advice to the designer, also has the capability to resolve conflicts. In contrast, the I3D system only offered advice to the designer.

7.3 System Architecture

The I3D+ system is organized in such a way that the data needed during the whole design process is retrieved from a central area. The requirements and other system parameters are stored in an Object Oriented Representation using COOL. The various expert agents also access these objects for their processing.

The overall architecture of the I3D+ system is shown in Figure 7.2. The expert agents are organized such that they have the capability to explicitly represent both domain knowledge, as well as negotiation knowledge. The circles represent the agent's domain knowledge, and the ellipses represent the agent's negotiation knowledge. The single-sided, solid line arrows indicate the data flows in the system. The double-sided, dotted line arrows indicate the negotiation possibilities between agents. The rectangle indicates the parameter value to be decided by the expert agents.

When there are two single-sided solid, line arrows pointing towards a rectangle, it shows a possible conflict situation; two expert agents can make decisions about the parameter in the rectangle. For example, Material Agent 1 (M1) and Material Agent 2 (M2) can make decision about the material parameter (M). When there is a single-sided, horizontal solid line arrow pointing towards an agent from a rectangle, it shows a different possible conflict situation; the agent could give negative criticism on the parameter value. For example the Material Critic (MC) can critique the value of the material parameter (M) decided by the Material Agent's, M1 and M2. In places where a double-sided, dotted line arrow connect two expert agents, it indicates that there is a possibility for the agents to get into negotiation.

Sections 7.3.1, 7.3.4, and 7.3.5, will explain in detail the negotiation architecture, communication architecture and the agent architecture respectively.

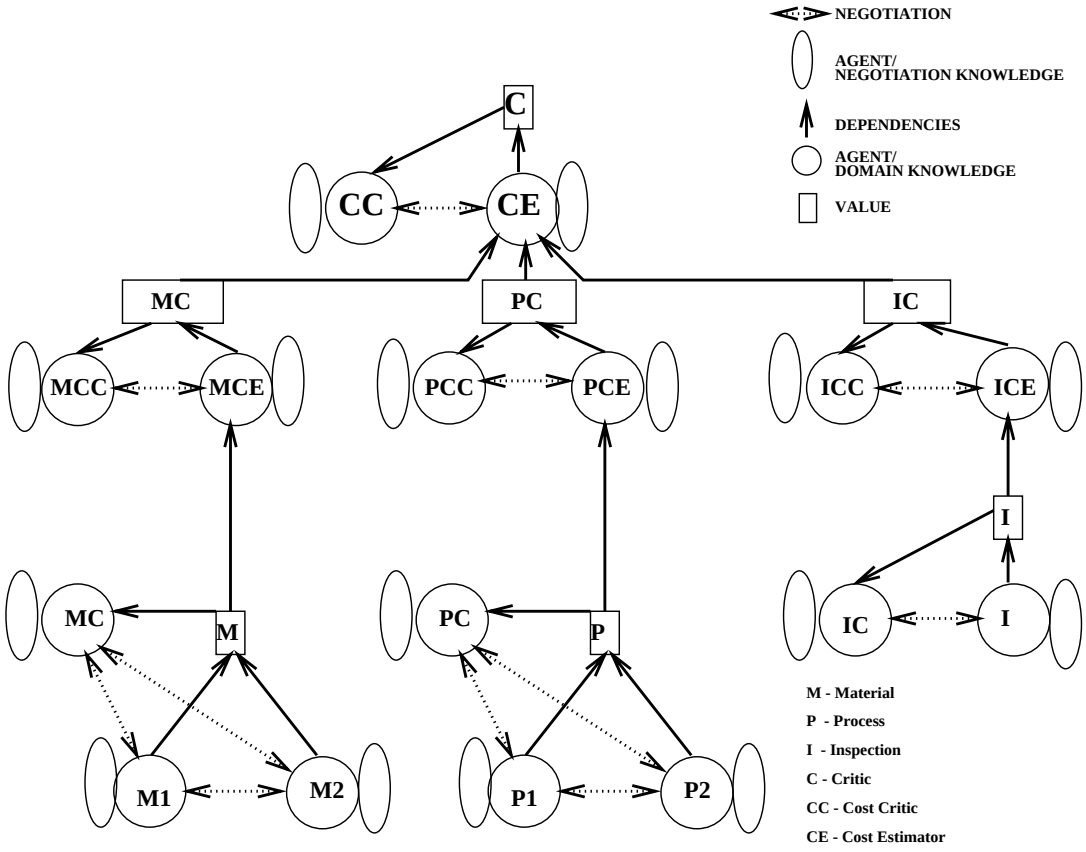

Figure 7.2: I3D+ System Architecture

7.3.1 Negotiation Architecture

Comparison of I3D+'s Negotiation Architecture with the attributes of negotiation are presented in Table 7.1. Each of these attributes are briefly discussed below.

In I3D+'s Negotiation Architecture, the computational model is based on agents exchanging their intentions and justifications. The agents are implemented as expert systems, using "C Language Integrated Production System"– (CLIPS). There is a no central monitor present in the system.

Attributes	**I3D+ System**
Computational Model	Based on exchanging intentions and justifications using CLIPS and COOL.
Central Monitor	NONE
Conflict Detection	Done by individual agents
Conflict Notification	Done by individual agents
Conflict Resolution	Done through exchange of intentions, and justification
Use of Design Rationale	NONE
Negotiation Mechanism	Iterative and done through exchange of intentions, and justifications
Negotiation Strategy	No centralized control, direct communication between agents
Evaluation Criteria	Comparison with the I3D System, as well as trying different conflict situations

Table 7.1: Attributes of Negotiation - I3D+ System

Each agent is capable of detecting any conflict situation and notifying the other agent involved in the conflict. Then the agents directly engage in the process of conflict resolution. During the process of conflict resolution, agents exchange their intentions and justifications.

The negotiation strategy is distributed such that the agents are directly involved in the negotiation process. The evaluation of the I3D+ system is done by comparing it with the results of the I3D system.

7.3.2 Conflict Situations

In this section, we will briefly discuss the various conflict situations that are possible in the I3D+ system as shown in Figure 7.3. In each situation two agents are involved in the negotiation process. All the agents implemented in the I3D+ system are *"single function agents"* [Brown 1993c]. A detailed discussion of the single function agent architecture is presented in section 7.3.6.

Agent that detects the conflict, takes the lead in negotiation
Global goal influences decision making

Conflict Situations	Agent 1 (Local Goal)	Agent 2 (Local Goal)	Global Goal(s) (Influence Decision)	Observation
1	X	Y	Z	Either one will win
2	X	Y	X	Agent 1 will win
3	X	Y	Y	Agent 2 will win
4	X	Y	X, Y	Either one will win
5	X	X	Z	Either one will win depending on their Knowledge (expertise)
6	X	X	X	Either one will win depending on their Knowledge (expertise)

Figure 7.3: Possible Conflict Situations

Each agent participating in the conflict situation has a local goal that it has to satisfy. Also, these agents have to try to meet a global goal at the same time as they are trying to satisfy their local goals. This scheme is based on the assumption that when trying to cooperatively solve a complex problem (global goal), there could be also many sub-problems (local goals) that have to be solved.

Some of these conflict situations were tried and their results discussed in section 8.3.

Situation 1

In the first situation, both the agents' goals are different from the global goal and hence either one of them could win. This situation is not implemented.

Situations 2 & 3

In the second and third situations, the global goal matches one of the local goals and hence, agent 1 wins in the second situation and agent 2 wins in the third situation. These situations are implemented.

Situation 4

In the fourth situation, the global goal has both the agents local goals. Hence either one of the agents could win depending on their expertise. This situation is not implemented.

Situations 5 & 6

In the fifth and sixth situations both the agents have the same goal, but the global goal could be the same or different. In such situations either one of the agents will again win depending on their expertise. These situations are implemented.

7.3.3 Negotiation Knowledge Representation

In order to achieve their goals each agent has its domain knowledge and negotiation knowledge represented explicitly. If there is no conflict between agents, only the domain knowledge is used. If there is a conflict, then the agents use both their domain and negotiation knowledge.

To illustrate this we will describe an example of a negotiation process between two agents involved in conflict situation 2.

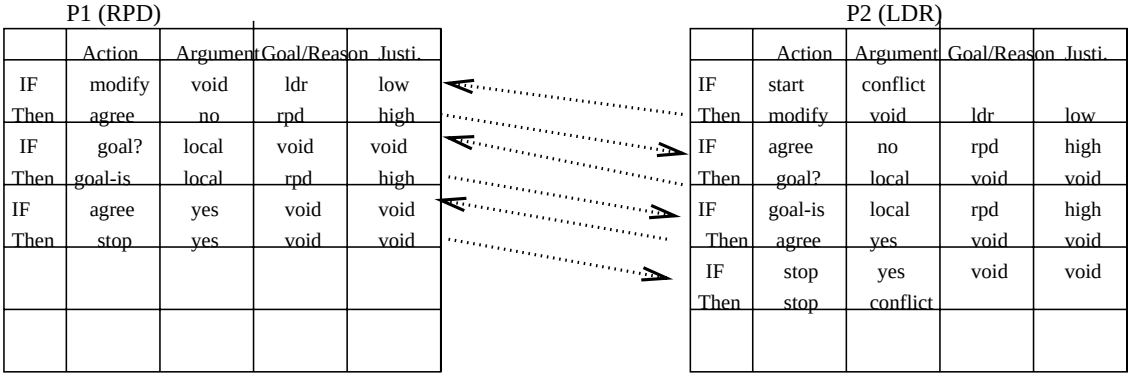

Figure 7.4: Negotiation Knowledge: Process Selection

In Figure 7.4, Agent 1 is the process selection agent 1 (P1), and agent 2 is the process selection agent 2 (P2). Both these agents decide the value of the process parameter. P1's goal is to increase the process attribute called Relative Percent Density (RPD). P2's goal is to increase another process attribute called Length to Diameter Ratio (LDR). The global goal is to have a high value for RPD.

Figure 7.4 shows how this negotiation knowledge is represented in the system for this conflict situation. The negotiation knowledge is represented as rules. Each *IF/THEN* pair represent a rule. The *IF* part represents the pre-conditions that must be satisfied for the rule to be activated. The *THEN* part represents the message to be sent if the pre-conditions are satisfied. As shown in Figure 7.4, initially P2 detects the conflict situation, and starts the negotiation process. This is indicated by the dotted arrow (message) going from table P2 to P1. The message sent is *(modify, void, ldr, low)*. Then P1 responds to P2's message. Thus the process continues back and forth till the conflict is resolved. This is indicated by the last rule in P2's table, where the negotiation process is stopped. In this situation, the process agent 1 eventually wins because its local goal matches the global goal.

7.3.4 Communication Architecture

The I3D+ system has a well defined communication architecture. The communication protocol, communication medium, and the communication language are clearly specified. The communication medium used for the communication between agents

is based on shared memory provided by the CLIPS Object Oriented Representation.

Attributes	**I3D+ System**
Communication Protocol	Based on Speech Acts
Communication Medium	Shared Memory
Communication Language	Common Communication Language

Table 7.2: Attributes of Communication - I3D+ System

The communication protocol used in the system, is derived from Speech Acts Theory [Austin 1962], [Searle 1969]. Speech Acts allow us to model people's intentions as a fixed set of actions. The message structure on which the communication language is based on is shown in Table 7.3. The message structure consists of: to whom the message is sent (addressee), from whom the message is received (speaker), and the message content. The message contains the following four primitives: action, argument, goal/reason, and justification. The primitives are: an action, e.g. agree, modify etc., an argument for the action, e.g. yes, no, increase, decrease, etc., a goal/reason, e.g. cost, strength, etc., and its justification, e.g. high, low, etc. The use of such a scheme allows the system to maintain a history of the other agent's dialog during the negotiation process.

Agents use a common language for interagent communication that consists of these primitive messages. This common language facilitates an easy means of communication between agents. This interagent language allows for the expression of agent intentions at some level of abstraction.

7.3.5 Agent Architecture

The expert agents in the I3D+ system are semi-autonomous. They are capable of resolving conflicts directly by negotiating with each other, but need assistance for scheduling. Due to this semi-autonomous nature, the grainsize of these expert agents is medium.

Message Content	**Semantic Expansion**
	$\prec x \succ$ non-terminal symbol | denotes alternatives
$\prec$ speech-act $\succ$	$\prec speaker \succ, \prec addressee \succ,$ $\prec action \succ, \prec argument \succ,$ $\prec goal/reason \succ, \prec justification \succ$
$\prec speaker \succ$	$\prec$ agent-name $\succ$
$\prec addressee \succ$	$\prec$ agent-name $\succ$
$\prec$ agent-name $\succ$	$M1, M2, P1, P2, I, MC, PC, IC, MCE, PCE, ICE,$ MCC, PCC, ICC, CC, CE
$\prec action \succ$	$agree\|modify\|goal?\|$ goal-is
$\prec argument \succ$	$yes\|no\|increase\|decrease\|local\|global\|void$
$\prec goal/reason \succ$	$cost\|tc\|hardness\|stress\|strength\|void$
$\prec justification \succ$	$high\|low\|void$

Table 7.3: Message Structure: Speech Acts based Communication Protocol

Attributes	**I3D+ System**
Agent Type	Semi-Autonomous
Agent Grainsize	Medium
Agent Domain Knowledge	Explicitly represented using Knowledge Bases
Agent Negotiation Knowledge	Explicitly represented using Knowledge Bases
Agent Privacy	Partial knowledge about other agents
Agent Examples	Material Selection Agents Process Selection Agents Inspection Selection Agent Cost Estimation Agents Critic Agents etc.

Table 7.4: Attributes of Agents - I3D+ System

The agents have the capability to represent explicitly both their domain knowledge and negotiation knowledge. Agents have partial knowledge about the other agents with whom they might have conflicts. There are a total of sixteen expert agents in the I3D+ system. Some examples of these expert agents are material selection, process selection, inspection selection, material criticism, cost estimation, etc.

7.3.6 Single Function Agents

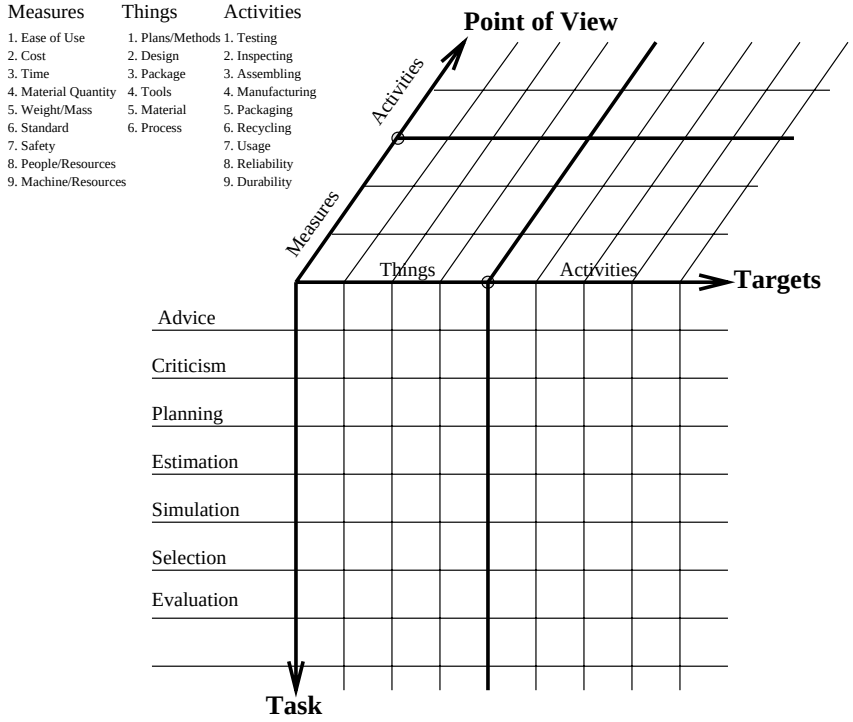

Figure 7.5: Single Function Agents: Task, Target, Point of View

The expert agents in the I3D+ System are implemented as *single function agents* [Brown 1993c]. The different kinds of single function agents are based on three dimensions. The first dimension is the agent's functionality, that is, what kind of a task it can perform, e.g., advice, criticism, estimation, etc. The second dimension is the agent's target domain, that is, what kind of a domain it is working in, e.g., material,

process, inspection, etc. The third dimension is the agent's point of view, e.g., cost, strength, safety, etc.

Figure 7.5 illustrates the possible combination of an agent's Task, Target and Point of View. Some examples of these single function agents implemented in the I3D+ System include, the material (target) selection (task) agent from the point of view of high thermal conductivity, material (target) selection (task) agent from the point of view of strength, process (target) selection (task) agent from the point of view of high Relative Percent Density, process (target) selection (task) agent from the point of view of high Length to Diameter Ratio, material (target) critic (task) from the point of view of cost; process (target) critic (task) from the point of view of cost, etc.

Not all the columns and rows in the three dimensional matrix shown in Figure 7.5 make sense. For example Selection (task) of a Usage (target) from the point of view of Weight does not make sense. The effect of these single function agents on the I3D+ System is discussed in section 8.3.

7.4 Scheduling Strategy

In comparison to the I3D system, where a fixed scheduling strategy was followed, the I3D+ system has a scheduling strategy based on *agenda*. The agenda driven scheduling strategy is highly flexible. Execution of the expert agents' are controlled by their priority ratings. By changing the priority ratings of the expert agents, we can totally control the sequence of their execution. This flexibility provides the possibility of experimentation with different sequences of execution of the design agents. Hence many, different possibilities of conflicts between agents and the subsequent negotiation process can be studied.

7.4.1 Agenda based Scheduler

An *Agenda* is a list of tasks a system could perform. Associated with each task there are usually two things: a list of reasons why the task is being proposed, often called justifications, and a computed priority rating factor for the task. The

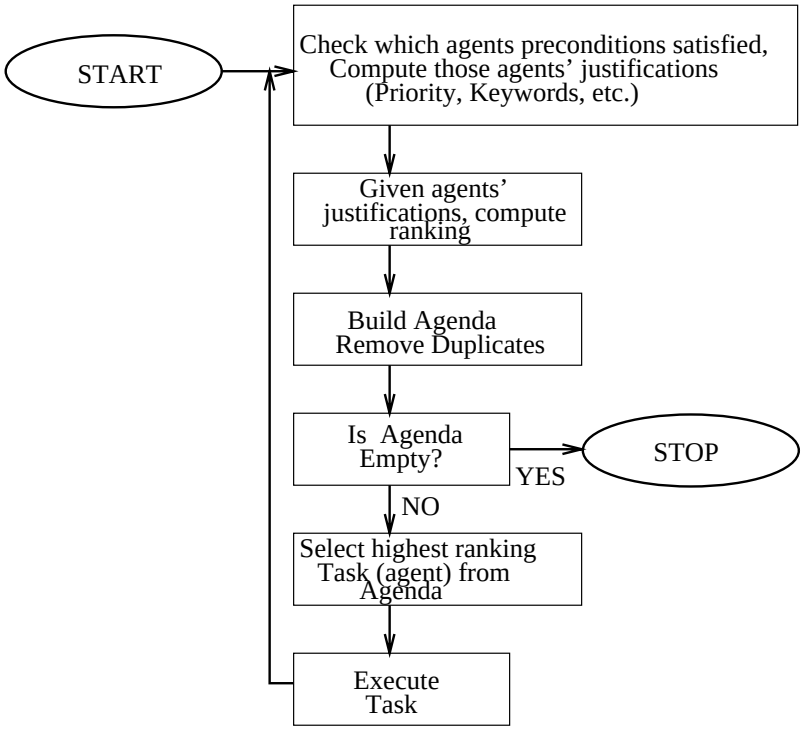

Figure 7.6: I3D+ System Scheduler

agenda based scheduler executes until the agenda is empty. It first chooses the most promising (highest priority) task from the agenda. It then executes the task. During execution of this task it may generate additional tasks. For every new task generated, the scheduler sees whether this new task is already in the agenda. If the task is already there on the agenda it takes no action. If the task is not there on the agenda, it inserts the task on the agenda. The scheduler continues until all the tasks on the agenda are executed. Once the agenda is empty the scheduler stops, bringing the system to a halt.

The agenda based scheduler implemented in the I3D+ system do not have all of the capabilities mentioned above. To keep the implementation simple, the scheduler only maintains the priority ratings for the tasks. It does not have a separate scheme to explicitly maintain the justifications. Here, the tasks are sorted by their priority rating. Hence inserting a new task means to find the appropriate place based on its rating and inserting it.

7.5 Implementation

This section presents the implementation of the I3D+ system. The three main issues addressed with regard to implementation are: the control flow, the data flow and the tools that were used. The two main tools that were used to implement the system are: C, in which all the system control functions, design and output were written; CLIPS, in which the expert systems were written. COOL, the object system part of CLIPS, in which the I3D+ system database was created.

7.5.1 Control Flow

There are two main aspects that are associated with the implementation of the control flow. The first aspect is to deal with the overall control of the system. The second aspect is to deal with the negotiation control of the system. The overall control of the system is done by the scheduler, which is implemented in C and shown in Figure 7.7. The user initially gives his requirements and specifications, asserted as facts to CLIPS. The scheduler then determines which design agents are capable of executing

and maintains a list of them. The list is sorted by the agents' priority rating. The agent with the highest priority is executed first. Subsequently, lower priority agents are executed. When an agent gets executed it updates the design database, using COOL.

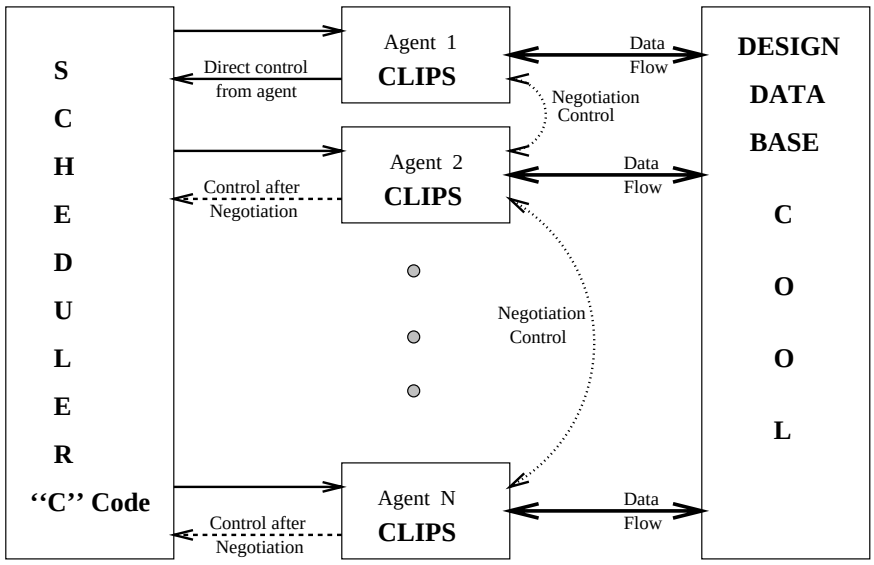

Figure 7.7: I3D+ System Control Flow

When an agent tries to update a particular parameter and finds out that the parameter value is already decided, it gets into a conflict situation. In this situation, the agent takes control from the scheduler and gets into the process of negotiation with the conflicting agent (i.e., the agent that last updated the parameter value). During the negotiation process, control is passed back and forth between the agents without the help of the scheduler, as shown in Figure 7.8. Once the conflict is resolved, control is then passed back to the scheduler by the agent who initially took control away from the scheduler.

In Figure 7.8, a detailed sequence between two negotiating agents is shown. The numbers indicate the control sequence. Initially, Material Agent 1 executes (1), decides the value for the material, and updates the database (2). It then returns control back to the scheduler (3). Eventually, Material Agent 2 gets scheduled and executes (4). When it tries to update the material value (5), it finds that Material Agent 1

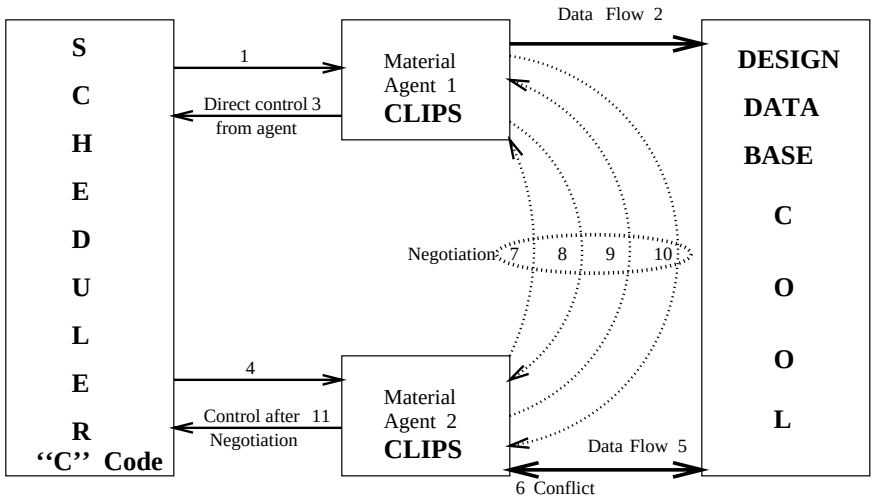

Figure 7.8: I3D+ System Negotiation Control Flow

has already decided the value (6). It discovers that the value it has decided is not the same as the value decided by Material Agent 1. This is a conflict situation. In order to resolve this conflict, it negotiates with Material Agent 1 (7,8,9,10). After the negotiation process is completed, Material Agent 2 returns control back to the scheduler (11).

7.5.2 Data Flow

The design data base and the data flow between the agents is shown in Figure 7.9. The design data base is implemented as a shared memory using the CLIPS Object Oriented Representation. Data is exchanged between the agents and the Object Oriented Representation using COOL.

The design agents decide the values of the parameters maintained in the design data base. The critic agents, when they get scheduled, look at these decisions and give their negative criticism. Data flows in the direction indicated by the arrows. In Figure 7.9, the arrows pointing towards the design data base indicate that the value is being decided by the agents. The arrows pointing towards the agents indicate that the agents use this value from the database for their processing.

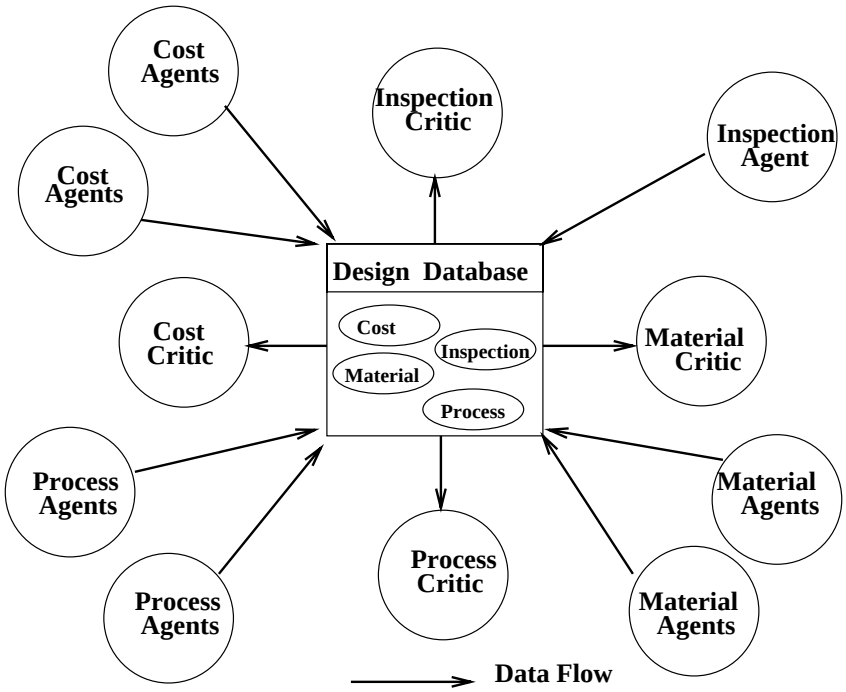

Figure 7.9: I3D+ System Data Flow

7.5.3 Implementation: General C Code

The functions of the C code include the scheduling of the different expert design agents, conflict detection, conflict notification and conflict resolution.

The greatest responsibility of the C code is as a link between the scheduler, CLIPS and COOL. It must convert the facts asserted by CLIPS into object instances in COOL. Agents, implemented using CLIPS, are able to detect, notify, and resolve conflicts using C functions.

7.5.4 Implementation: CLIPS and COOL

CLIPS was used to build the expert systems which represent the various types of agents. COOL is used to act as an object-oriented design database.

All of the rules written in CLIPS operate by attempting to match facts in the condition part of each rule to those in the fact-list (user requirements) and then to execute an action when those facts are matched. The execution of CLIPS is con-

trolled from outside of CLIPS. The C code has to be invoked to send a "RunCLIPS" command to execute the rules.

The COOL database contains all of the objects that can exist during the design. Each object has a number of attributes. These objects in the database act as classes. When a design is performed, instances of these classes are created which have specific values for their attributes. By referencing the instances of these objects, the agents can get a detailed view of the current state of the design.

7.6 Summary

This chapter examined some of the details of the implementation of the I3D+ system. It also showed some details about CLIPS, COOL and the C code that holds the I3D+ system together. A view of the control structure for the expert system agents was shown. The next chapter is an evaluation of the system, results, conclusions, and suggestions for future work.

Chapter 8

Conclusions

8.1 Introduction

This chapter discusses how well this thesis meets the goals described in Section 1.2. It also discusses the evaluation performed on the I3D+ Negotiation System. These two points are covered in Section 8.2 below. Section 8.3 presents the results of the evaluation of the I3D+ Negotiation System. Future extensions to this system, as well as to conflict resolution and negotiation in general, are discussed in Section 8.5.

8.2 Evaluation Criteria

Evaluation of the I3D+ Negotiation System was done based on two important criteria.

The first criterion for evaluation, was based on how well the I3D+ system produces the *same results* as the I3D system.

The second criterion for evaluation is based on the human expert's opinion of the solution produced by the I3D+ System. Some of the aspects that were evaluated were, the degree to which the dialogue is understandable by the user, the degree to which the dialogue is realistic, the degree to which the knowledge represented in the

agents is readable, and the correctness of the results.

8.3 Evaluation

In this Section we will present the results obtained from the evaluation of the I3D+ negotiation system.

To evaluate I3D+ based on the first criterion, the results of the negotiation-based system were compared with the original I3D system. This comparison showed that the I3D+ system produced the *same results* as the I3D system. By this process we were also able to evaluate the capabilities of the I3D+ negotiation system. The I3D+ system had the capability to detect and notify conflicts, and to resolve these conflicts through the process of negotiation along with providing advice to the designer. The I3D system did not have any capability to represent conflicts, and it only gave advice to the designer.

For example, in the I3D+ system we can simulate a typical test case of functional requirements which will prompt the material selection expert agents to decide on two different choice of materials, leading to the process of negotiation between them to resolve this conflict. Another test case involves the process critic to criticise the choice of manufacturing process selected by the process selection expert agents. This situation also leads to negotiation to resolve the conflict between the process critic and the process selection agents. Appendix 9 lists the general trace of the I3D System, simulated using the I3D+ System. Appendix 10 lists the general trace of the I3D+ System. Comparison of these two traces demonstrates the negotiating capabilities of the I3D+ system. Appendix 11 lists the trace of the I3D+ system collected during different conflict situations.

Based on the second criterion, the opinion of two human experts in the field of Manufacturing Engineering was sought. The first expert is a Graduate Student working towards his master's program in Manufacturing Engineering at Worcester Polytechnic Institute. The second expert is a Professor in the Manufacturing Engineering Department at Worcester Polytechnic Institute.

The experts felt that the negotiation dialogue produced by these expert agents is understandable. They mentioned that the dialogue looked similar to how human

experts would argue and convince each other of their respective point of view. They also expressed the view that the dialogue is realistic. They also had a look at how the knowledge is represented in these expert agents, in the form of rules, and found those rules readable.

8.3.1 Quantitative Evaluation

The execution time of the I3D+ system was compared with the I3D system. The I3D system took about 11 Seconds to execute through both the conceptual and detailed design phase. The I3D+ system took about 20 Seconds for execution because it had also to resolve conflicts during the process of its execution. If more conflicts were introduced the execution time of the I3D+ system increased to about 32 Seconds. Figure 8.1 shows the Execution Time versus Number of Conflicts performance of the I3D+ system.

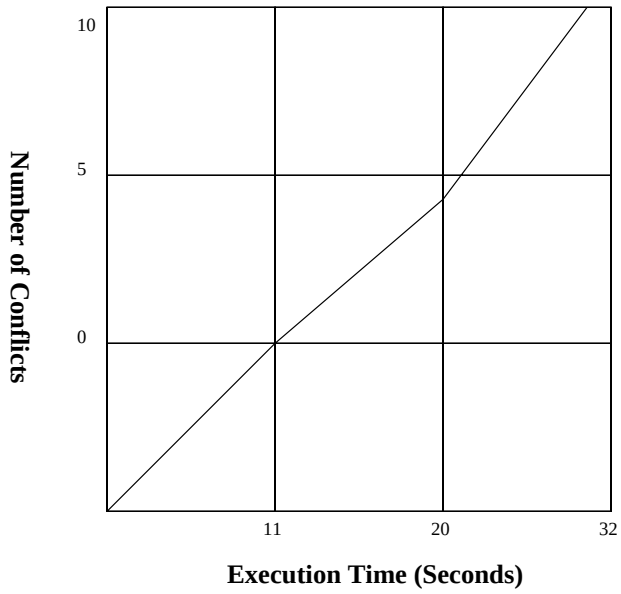

Figure 8.1: Execution Time versus Number of Conflicts in I3D+

Similarly the complexity of the I3D+ system was compared with the I3D system. Since the I3D+ system had the capability to negotiate and resolve conflicts, it had negotiation knowledge represented in the form of rules. Each expert agent on the

average had about 20 rules to represent the negotiation knowledge. The system had a total of about twenty-two expert agents. Hence there were about *22 X 18 = 396* extra rules in the I3D+ system compared to the I3D system.

Also, the I3D+ system had negotiation dialogue between the Cost Critic (CC), Material Cost Critic (MCC) , Process Cost Critic (PCC), Cost Estimator (CE) Material Cost Estimator (MCE), Process Cost Estimator (PCE), Material Critic (MC), Material Agent 1 (M1), Material Agent 2 (M2), Process Critic (PC), Process Agent 1 (P1), Process Agent 2 (P2), Inspection Critic (IC), and Inspection Agent (I). On the average when two of these agents negotiate they produce 8 messages back and forth. We have a total of 12 agents that negotiate with each other in a typical run, producing potentially *12 X 8 = 96* messages. The results (number of messages) from three test cases are given below. In the first case, when there are no conflicts, the system only gives advice to the user. In the second case, when there are 5 conflicts, the system generates about 45 messages. In the third case, when there are 10 conflicts, the system generates about 95 messages.

8.4 Observation

The use of single function agents in the I3D+ system to implement the expert agents leads to some interesting observations. Though the single function agents were responsible for precisely defining each agents functionality and a very focussed negotiation dialogue, they had a couple of drawbacks.

Since each agent was broken down to do a single function, there were a large number of agents in the system and hence the message overhead was high. If we had large agents (multi-function), then there would have been fewer agents, and hence fewer messages. Also in large agents the conflicts are buried in them, hence less amount of conflicts between them and hence less negotiation (i.e., less messages).

In the I3D+ system there were about twenty-two single function agents. Hence finding a solution among them may result in a local maxima, instead of the optimal solution. This is because, if this system is modeled as a constraint satisfaction problem, the time taken to propagate the constraints and to do a exhaustive search to find the optimal solution will be prohibitively expensive. It is also more difficult to

transfer constraint information between many agents.

8.5 Future Work

There are two conflict situations 1 & 4 in Figure 7.3, which have not been implemented but could be implemented. This was due to time constraints. There is the potential for around thirty agents in the system as it stands, but only twenty-two agents are included. The negotiation knowledge of these agents can be expanded by adding more negotiation rules. This will help to produce a more detailed dialogue between agents during the process of negotiation. The evaluation of the single function agents is not yet very conclusive. More research is needed in this area to discover their strengths and limitations.

The system currently takes care of negotiation between two agents at a time. More work is needed to handle negotiation between more than two agents. Also, the negotiation history is not captured to aid future negotiations. Research in these areas can be found in [Klein 1993a]. Also, the agenda mechanism implemented in the system can be extended.

A full-blown system may use multiple processes on multiple machines at different locations. This is because human experts can be distributed at different locations. There are many ways in which this system could be expanded. However, I3D+ is a complete system, and needs no major improvements to fulfill is task.

8.6 Summary

This thesis has shown that there is a large amount of work being done in Negotiation and Conflict Resolution, and that there is a need to bring that research into practical use. The I3D+ Negotiation System is one such foray into this area. The I3D+ System is a real Concurrent Engineering System in the domain of manufacturing and materials engineering, with negotiation capabilities. Though much has been accomplished there is still room for expansion and improvement. Negotiation is a growing area of interest in the domain of Cooperative Problem Solving.

Bibliography

[Altamuro 1991] V. M. Altamuro. Strategic Product Design. *Concurrent Engineering,* Vol. 1, No. 2, 1991, pp. 39-45.

[Austin 1962] J. L. Austin. *How to Do Things With Words,* Cambridge, MA: Harvard University Press, and London: Clarendon Press, 1962.

[Bedworth et al 1991] D. D. Bedworth, M. R. Henderson, P. M. Wolfe. *Computer-Integrated Design and Manufacturing,* McGraw-Hill, New York, NY, 1991, pp. 134-176.

[Bond & Gasser 1992] A. H. Bond, L. Gasser. A Subject-Indexed Bibliography of Distributed Artificial Intelligence. *IEEE Transactions on Systems, Man and Cybernetics,* Vol. 22, No. 6, November/December 1992, pp 1260-1281.

[Brown 1985] D. C. Brown. Failure Handling in a Design Expert System. *Computer-Aided Design, special issue,* (Ed.) J.S.Gero, Butterworths, November 1985.

[Brown 1992] D. C. Brown. Design. *Encyclopedia of Artificial Intelligence,* 2nd edn., S. C. Shapiro (Ed.), J. Wiley, 1992.

[Brown et al 1993a] D. C. Brown, R. E. Douglas, Jr. & D. C. Zenger. A Concurrent Engineering Demonstration System for

use in Training Engineers and Managers. *AI in Engineering Conference,* 1993.

[Brown & Douglas 1993b] D. C. Brown & R. E. Douglas, Jr. Concurrent Accumulation of knowledge: A View of CE. *The Handbook of Concurrent Design and Manufacturing,* (Eds.) H.R.Parsaei & W.G.Sullivan, Chapman & Hall, 1993, pp. 402-412.

[Brown 1993c] D. C. Brown. Exploring the Role of Single Function Agents in Negotiation. *AAAI-93 Workshop on AI in Collaborative Design,* Washington, D.C., July 1993.

[Giarratano 1991a] J. C. Giarratano. *CLIPS User Guide,* Volume 1, Rules. NASA Lyndon B. Johnson Space Center Information Systems Directorate, Software Technology Branch, Version 5.0, May 1991.

[Giarratano 1991b] J. C. Giarratano. *CLIPS User Guide,* Volume 2, Objects. NASA Lyndon B. Johnson Space Center Information Systems Directorate, Software Technology Branch, Version 5.0, May 1991.

[Bussmann & Muller 1992] S. Bussmann, J. Muller. A Negotiation Framework for Cooperating Agents. *CKBS Keele,* 1992.

[Bussmann & Muller 1993a] S. Bussmann, J. Muller. A Communication Architecture for Cooperating Agents. *Journal on Computers and Artificial Intelligence,* January 1993.

[Bussmann & Muller 1993b] S. Bussmann, J. Muller. Bargaining Agents. *Submitted to 14th International Joint Conference on Artificial Intelligence,* 1993.

[Davis & Smith 1983] R. Davis, R.G. Smith. Negotiation as a metaphor for distributed problem solving. *Artificial Intelligence,* 20 (1), 1983, pp. 63-109.

[Jagannathan et al 1991] V. Jagannathan, K. J. Cleetus, R. Kannan, A.S.Matsumoto, J. W. Lewis. Computer Support for Concurrent Engineering. *Concurrent Engineering,* Vol. 1, No. 5, 1991, pp. 14-30.

[Kannapan & Marshek 1991] S. M. Kannapan, K. M. Marshek. A Schema for Negotiation between Intelligent Design Agents in Concurrent Engineering. *Intelligent Computer Aided Design,* (Eds.) D.C.Brown, M.Waldron & H.Yoshikawa, Elsevier Science Publishers B.V. (North-Holland), July 1992, pp. 1-26.

[Kannapan et al 1992] S. M. Kannapan, D. G. Bell, D. L. Taylor. Structuring and Coordinating Information in Product Development. *Xerox Design Research Institute Technical Repor,* Cornell University, 1992.

[Katz 1990] R. H. Katz. Towards a Unified Framework for Version Modeling in Engineering Databases. *ACM Computing Surveys,* Vol. 22, No. 4, 1990, pp. 375-408.

[Klein 1990] M. Klein. Conflict Resolution in Cooperative Design. *The International Journal for Artificial Intelligence in Engineering,* Vol. 4, No. 4, 1990, pp. 168-180.

[Klein 1991] M. Klein. Supporting Conflict Resolution in Cooperative Design Systems. *IEEE Transactions on Systems, Man and Cybernetics,* Vol. 21, No. 6, November/December 1991.

[Klein 1993a] M. Klein. Capturing Design Rationale in Concurrent Engineering Teams. *IEEE Computer. Special Issue on Computer Support for Concurrent Engineering,* January 1993.

[Klein 1993b] M. Klein. Supporting Conflict Management in Cooperative Design Teams. *Journal of Group Decision Making and Negotiations. Special Issue on the 1992 Distributed Artificial Intelligence Workshop,* March 1993.

[Kott et al 1991] A. Kott, C. Kollar, A. Cederquist. The Role of Product Modeling in Concurrent Engineering Environments. Article submitted to the *Journal of Systems Automation,* 1991.

[Lander & Lesser 1989] S. E. Lander, V. R. Lesser. A Framework for Cooperative Problem-Solving Among Knowledge-Based Systems. *Technical Report, Department of Computer and Information Science, University of Massachusetts* Amherst, MA 01003, 1989.

[Lander & Lesser 1991a] S. E. Lander, V. R. Lesser. Customizing Distributed Search Among Agents with Heterogeneous Knowledge. *Technical Report, Department of Computer and Information Science, University of Massachusetts* Amherst, MA 01003, 1991.

[Lander et al 1991b] S. E. Lander, V. R. Lesser, M. E. Connell. Knowledge-based conflict resolution for cooperation among expert agents. *Computer Aided Cooperative Product Development,* (Eds.) D. Sriram, R. Logcher & S. Fukuda, Lecture Notes Series, No. 492, Springer Verlag, 1991, pp. 253-268.

[Lander & Lesser 1992] S. E. Lander, V. R. Lesser. Negotiated Search: Organizing cooperative search among heterogeneous expert agents. *Proceedings of the Fifth International Symposium on Artificial Intelligence, Applications in Manufacturing and Robotics* Cancun, Mexico, December 1992.

[Lemon et al 1990] J. R. Lemon, W. E. Dacey, E. J. Carl. Concurrent Product/Process Development. International TechneGroup Incorporated Technical Report, 1990.

[Londoño et al 1991] F. Londoño, K. J. Cleetus, Y. V. Reddy. A Blackboard Scheme for Cooperative Problem-Solving by Human Experts. *Computer Aided Cooperative Product Development,* (Eds.) D. Sriram, R. Logcher & S. Fukuda, Lecture Notes Series, No. 492, Springer Verlag, 1991.

[Nii 1986a] H. P. Nii. The Blackboard Model of Problem Solving and the Evolution of Blackboard Architectures. *The AI Magazine,* Summer 1986, pp. 38-53,.

[Nii 1986b] H. P. Nii. Blackboard Application Systems and a Knowledge Engineering Perspective. *The AI Magazine,* August 1986, pp. 82-107.

[Rosenschein & Genesereth 1985] J. S. Rosenschein, M.R. Geesereth. Deals among rational agents. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence,* Los Angeles, California, August 1985, pp. 91-99.

[Searle 1969] J. Searle. *Speech Acts,* Cambridge: Harvard University Press, 1969.

[Smith & Davis 1988] R. G. Smith, R. Davis. Frameworks for Cooperation in Distributed Problem Solving. In A. H. Bond, L. Gasser, editors, *Readings in Distributed Artificial Intelligence,* Morgan Kaufmann, 1988, pp. 61-71.

[Smith 1980] R. G. Smith. The Contract Net Protocol: High-Level Communication and control in a Distributed Problem Solver. *IEEE Transactions on Computers,* C-29 (12), 1980.

[Sriram et al 1989] D. Sriram, R. D. Logcher, N. Groleau, J. Cherneff. DICE: An Object Oriented Programming Environment For Cooperative Engineering Design. *Technical Report No: IESL-89-03,* Mass. Institute of Technology, Cambridge, MA, 1989.

[Stoll 1986] H. W. Stoll. Design for Manufacture: an overview, 1986. Reprinted in *Design for Manufacture,* J. Corbett, M. Dooner, J. Meleka, C. Pym, eds., Addison-Wesley, Reading, MA, 1991, pp. 107-129.

[Subramanian et al 1990] E. Subramanian, G. Podnar, A. Westerberg. A Shared Computational Environment for Concurrent Engineering. *Engineering Design Research Center Technical Report,* Carnegie Mellon University, Pittsburgh, PA, 1990.

[Sycara 1985] S. P. Sycara. Persuasive Argumentation in Resolution of Collective Bargaining Impasses. *Proceedings of the Seventh Annual Conference of the Cognitive Science Society,* Irvine, California, August 1985, pp. 356-360.

[Sycara 1987] S. P. Sycara. Finding Creative Solutions in Adversial Impasses. *Proceedings of the Ninth Annual Confer-*

ence of the Cognitive Science Society, Seattle, Washington, July 1987.

[Sycara 1988a] S. P. Sycara. Utility Theory in Conflict Resolution. *Annals of Operations Research,* Vol. 12, J.C. Baltzer AG, Scientific Publishing Company, 1988, pp. 65-84.

[Sycara 1988b] S. P. Sycara. Resolving Goal Conflicts via Negotiation. *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88),* Vol. I, St. Paul, Minnesota, August 1988, pp. 245-250.

[Sycara 1988c] S. P. Sycara. Patching Old Plans. *Proceedings of the Tenth Annual Conference of the Cognitive Science Society,* 1988.

[Sycara 1990a] S. P. Sycara. Persuasive Argumentation in Negotiation. *Theory and Design,* Kluwer Academic Published, Printed in the Netherlands, 1990, 28: 203-242.

[Sycara 1990b] S. P. Sycara. Cooperative Negotiation in Concurrent Engineering Design. *Cooperative Engineering Design,* Springer Verlag Publications, 1990.

[Sycara 1990c] S. P. Sycara. Negotiation Planning: An AI Approach. *European Journal of Operations Research,* Vol. 46, Elsevier Science Publishers B.V., North-Holland, 1990, pp. 216-234.

[Sycara 1991] S. P. Sycara. Negotiation in Concurrent Engineering Design. *Computer Aided Cooperative Product Development,* (Eds.) D. Sriram, R. Logcher & S. Fukuda, Lecture Notes Series, No. 492, Springer Verlag, 1991, pp. 269-297.

[Velthuijsen & Griffeth 1992] H. Velthuijsen and N. D. Griffeth. Negotiation in Telecommunications Systems. *National Workshop on Artificial Intelligence, AAAI,* July 1992.

[Victor et al, 1993] S. K. Victor, D. C. Brown, J. J. Bausch, D. C. Zenger, R. Ludwig. R. D. Sisson. Using Multiple Expert Systems with Distinct Roles in a Concurrent Engineering System for Powder Ceramic Components. *International Conference on Artificial in Engineering,* Tolouse, France, July 1993.

[Werkman 1990] K. J. Werkman. Knowledge-based model of negotiation using shareable perspectives. In M. Huhns, editor, *10th International Workshop on Distributed Artificial Intelligence,* Bandera, Texas, 1990.

[Werkman & Barone 1991] K. J. Werkman, M. Barone. Evaluating Alternative Connection Designs Through Multiagent Negotiation. *Computer Aided Cooperative Product Development,* (Eds.) D. Sriram, R. Logcher & S. Fukuda, Lecture Notes Series, No. 492, Springer Verlag, 1991, pp. 298-333.

[Werkman 1992] K. J. Werkman. Multiple Agent Cooperative Design Evaluation using Negotiation. *Second International Conference on Artificial Intelligence in Design,* Pittsburg, PA, June 1992.

[Zenger et al 1993] D. C. Zenger, D. C. Brown, J. J. Bausch III, R. D. Sisson, Jr., R. Ludwig. An interactive design and manufacturing methodology for the conceptual design phase. *CMSE's 7th Conference with Industry,* Lehigh University, Bethlehem, PA, May 1993.

[Zlotkin & Rosenschein 1989] G. Zlotkin, J. S. Rosenschein. Negotiation and task sharing among autonomous agents in cooperative domains. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence,* Detroit, Michigan, August 1989, pp. 912-917.

[Zlotkin & Rosenschein 1990] G. Zlotkin, J. S. Rosenschein. Negotiation and conflict resolution in non-cooperative domains. *Proceedings of the Eighth National Conference on Artificial Intelligence,* Boston, Massachusetts, July 1990, pp. 100-105.

[Zlotkin & Rosenschein 1992] G. Zlotkin, J. S. Rosenschein. A Domain Theory for Task Oriented Negotiation. In A. Cesta, R. Conte, and M. Miceli, editors, *Pre-Proceedings of the Fourth European Workshop on Modeling Autonomous Agents in a Multi-Agent World,* San Cimino, Italy, August 1992.

Appendix A

Traces from I3D+ Negotiation System: Without Conflicts

This section presents the execution of the I3D+ system without conflicts. In this case, it produces the same results as the I3D system. This trace shows the system offering only advice to the designer.

```
##################################################

   Agents in AGENDA with their priority rating

##################################################
m1 199
p1 197
i 170
mce 160
pce 155
ce 140
mced 30
pced 20
iced 10
ced 5
```

```
################################################

       Entering    SCHEDULING     Loop

################################################

SCHEDULING AGENT: m1 With PRIORITY: 199

Conceptual Design    ##############################
          E X P E R T      S Y S T E M
                    F O R
    M A N U F A C T U R I N G    P L A N N I N G
      ( M A T E R I A L    S E L E C T I O N  1)
################################################

The material selected is

   Zirconia

SCHEDULING AGENT: p1 With PRIORITY: 197

Conceptual Design    ##############################
          E X P E R T      S Y S T E M
                    F O R
    M A N U F A C T U R I N G    P L A N N I N G
     ( P R O C E S S      S E L E C T I O N  1)
################################################

The Process Selected is

   Uniaxial Dry Pressing
```

```
SCHEDULING AGENT: i With PRIORITY: 170

Conceptual Design    ##############################
          E X P E R T      S Y S T E M
                     F O R
       I N S P E C T I O N   A D V I C E
##################################################

The inspection method selected is

  Ultrasonic Imaging : C-Scan

Ceramics Applicability
 GREEN   : Low probability of successful
           application to Ceramics
 SINTERED: Under development for use with Ceramics

Advantages
Fast
Sensitive to Cracks
Disadvantages
Requires a Coupling Agent

SCHEDULING AGENT: mce With PRIORITY: 160

Conceptual Design    ##############################
          E X P E R T      S Y S T E M
                     F O R
M A T E R I A L   C O S T    E S T I M A T I O N
##################################################
The MATERIAL selected was:       ZIRCONIA
```

```
     Material Cost   $  0.01 Per Part

            For a  Batch Size of 1000 Parts

SCHEDULING AGENT: pce With PRIORITY: 155

Conceptual Design    ############################
           E X P E R T      S Y S T E M
                      F O R
P R O C E S S      C O S T     E S T I M A T I O N
##################################################
The PROCESS selected was:     UNIAXIAL DRY PRESSING

     Process Cost    $  0.91 Per Part

     Total Tooling Cost is   $20000.00

            For a  Batch Size of 1000 Parts

SCHEDULING AGENT: ce With PRIORITY: 140

Conceptual Design    ############################
           E X P E R T      S Y S T E M
                      F O R
         C O S T     E S T I M A T I O N
##################################################

Based on information from other Cost Estimators

    and for a Batch Size of 1000 Parts

           the Total Cost  is $ 20.92 Per Part
```

```
SCHEDULING AGENT: mced With PRIORITY: 30

Detailed   Design    ############################
           E X P E R T      S Y S T E M
                      F O R
M A T E R I A L    C O S T     E S T I M A T I O N
################################################
The MATERIAL selected was:       ZIRCONIA

     Material Cost   $  0.01 Per Part

            For a  Batch Size of 1000 Parts

SCHEDULING AGENT: pced With PRIORITY: 20

Detailed   Design    ############################
           E X P E R T      S Y S T E M
                      F O R
P R O C E S S      C O S T     E S T I M A T I O N
################################################
The PROCESS selected was:     UNIAXIAL DRY PRESSING

     Primary Process Cost    $  0.91 Per Part

     Secondary Process Cost  $  0.00 Per Part

     Tooling Cost            $ 20000.0

            For a  Batch Size of 1000 Parts
Note :
```

```
   Secondary Process was not selected.

SCHEDULING AGENT: iced With PRIORITY: 10

Detailed Design   ################################
          E X P E R T     S Y S T E M
                    F O R
I N S P E C T I O N  C O S T  E S T I M A T I O N
################################################
The INSPECTION method selected was:      C Scan

     Inspection Cost         $ 35.00 Per Part

           For a  Batch Size of 1000 Parts

SCHEDULING AGENT: ced With PRIORITY: 5

Detailed Design   ################################
          E X P E R T     S Y S T E M
                    F O R
        C O S T    E S T I M A T I O N
################################################

Based on information from other Cost Estimators

    and for a Batch Size of 1000 Parts

          the Total Cost  is $ 55.92 Per Part
```

Appendix B

Traces from I3D+ Negotiation System: With Conflicts

This section presents the execution of the I3D+ system with conflicts. This trace shows that the system, along with offering advice to the designer, also resolves conflicts through negotiation.

```
##################################################

   Agents in AGENDA with their priority rating

##################################################
m1 199
p1 197
m2 195
p2 190
mc 185
pc 180
i 170
mce 160
pce 155
ce 140
```

```
mcc 130
mced 30
pced 20
iced 10
ced 5
##################################################

       Entering    SCHEDULING     Loop

##################################################

SCHEDULING AGENT: m1 With PRIORITY: 199

Conceptual Design    ##############################
          E X P E R T      S Y S T E M
                     F O R
    M A N U F A C T U R I N G    P L A N N I N G
      ( M A T E R I A L    S E L E C T I O N  1)
##################################################
NO CONFLICT in material selection...
......even though material was already decided!

The material selected is

   Zirconia

SCHEDULING AGENT: p1 With PRIORITY: 197

Conceptual Design    ##############################
          E X P E R T      S Y S T E M
                     F O R
    M A N U F A C T U R I N G    P L A N N I N G
```

```
    ( P R O C E S S     S E L E C T I O N  1)
##################################################
NO CONFLICT in process selection...
......even though process was already decided!

The Process Selected is

   Uniaxial Dry Pressing

SCHEDULING AGENT: m2 With PRIORITY: 195

Conceptual Design    ############################
          E X P E R T     S Y S T E M
                    F O R
    M A N U F A C T U R I N G    P L A N N I N G
      ( M A T E R I A L    S E L E C T I O N  2)
##################################################
CONFLICT in material selection...
..................... Starting NEGOTIATION

The material selected is

   Silicon Carbide

==================================================
MATERIAL AGENT 2: NEGOTIATION Started.............
               Control taken away from SCHEDULER.
==================================================
MATERIAL AGENT 2:  Change the material because strength is Low
MESSAGE: (modify, void, strength, low)
Material Agent 2
    Sending Message to Material Agent 1 ..........
```

```
........Negotiation continuing.............
MATERIAL AGENT 1:  Though the strength is low, material has high TC!
MESSAGE: (agree, no, tc, high)
Material Agent 1
    Sending Message to Material Agent 2 ..........

........Negotiation continuing.............
MATERIAL AGENT 2:  What is your Local Goal ?
MESSAGE: (goal?, local, void, void)
Material Agent 2
    Sending Message to Material Agent 1 ..........

........Negotiation continuing.............
MATERIAL AGENT 1:  My local goal is to increase TC
MESSAGE: (goal-is, local, tc, high)
Material Agent 1
    Sending Message to Material Agent 2 ..........

........Negotiation continuing.............
MATERIAL AGENT 2:  I agree to your decision of material choice
    because your goal matches the global goal.
MESSAGE: (agree, yes, void, void)
Material Agent 2
    Sending Message to Material Agent 1 ..........

........Negotiation continuing.............
MATERIAL AGENT 1:  Yes! I agree too...........
  I am giving back control to you.
MESSAGE: (stop, yes, void, void)
Material Agent 1
    Sending Message to Material Agent 2 ..........
```

```
================================================
MATERIAL AGENT 2: ............Halted NEGOTIATION.
                 Control Given back to SCHEDULER.
################################################

SCHEDULING AGENT: p2 With PRIORITY: 190

Conceptual Design    ############################
          E X P E R T      S Y S T E M
                     F O R
    M A N U F A C T U R I N G    P L A N N I N G
     ( P R O C E S S      S E L E C T I O N  2)
################################################
CONFLICT in process selection...
.................... Starting NEGOTIATION

The Process Selected is

   Uniaxial Hot Pressing

================================================
PROCESS  AGENT 2:  NEGOTIATION Started..........
               Control taken away from SCHEDULER.
================================================
PROCESS  AGENT 2:  Change the process because LDR is Low
MESSAGE: (modify, void, LDR, low)
Process Agent 2
    Sending Message to Process Agent 1 ..........

.........Negotiation continuing.............
PROCESS AGENT 1:  Though the LDR is low, Part has high RPD
```

```
MESSAGE: (agree, no, RPD, high)
Process Agent 1
    Sending Message to Process Agent 2 ..........

.........Negotiation continuing.............
PROCESS AGENT 2:  What is your Local Goal ?
MESSAGE: (goal?, local, void, void)
Process Agent 2
    Sending Message to Process Agent 1 ..........

.........Negotiation continuing.............
PROCESS  AGENT 1:  My local goal is to have high RPD
MESSAGE: (goal-is, local, RPD, high)
Process Agent 1
    Sending Message to Process Agent 2 ..........

.........Negotiation continuing.............
PROCESS  AGENT 2:  I agree to your decision of process  choice
    because your goal matches the global goal.
MESSAGE: (agree, yes, void, void)
Process Agent 2
    Sending Message to Process Agent 1 ..........

.........Negotiation continuing.............
PROCESS  AGENT 1:  Yes! I agree too...........
  I am giving back control to you.
MESSAGE: (stop, yes, void, void)
Process Agent 1
    Sending Message to Process Agent 2 ..........

================================================
PROCESS  AGENT 2:.............Halted NEGOTIATION.
```

```
                Control Given back to SCHEDULER.
##################################################

SCHEDULING AGENT: mc With PRIORITY: 185

Conceptual Design    ############################
          E X P E R T      S Y S T E M
                     F O R
       M A T E R I A L    C R I T I C I S M
##################################################
CRITICISM of Material Selection due to HIGH COST
.............................Starting NEGOTIATION

==================================================
MATERIAL CRITIC : NEGOTIATION Started............
               Control taken away from SCHEDULER.
==================================================
MATERIAL CRITIC :  Change the material because cost is high
MESSAGE: (modify, void, cost, high)
Material Critic
    Sending Message to Material Agent 1 ..........

........Negotiation continuing.............
MATERIAL AGENT 1:  Though the cost is high, material has high TC!
MESSAGE: (agree, no, tc, high)
Material Agent 1
    Sending Message to Material Critic  ..........

........Negotiation continuing.............
MATERIAL CRITIC :  What is your Local Goal ?
MESSAGE: (goal?, local, void, void)
Material Critic
```

```
    Sending Message to Material Agent 1 ..........

.........Negotiation continuing.............
MATERIAL AGENT 1:  My local goal is to increase TC
MESSAGE: (goal-is, local, tc, high)
Material Agent 1
    Sending Message to Material Critic  ..........

.........Negotiation continuing.............
MATERIAL CRITIC :  I agree to your decision of material choice
    because your goal matches the global goal.
MESSAGE: (agree, yes, void, void)
Material Critic
    Sending Message to Material Agent 1 ..........

.........Negotiation continuing.............
MATERIAL AGENT 1:  Yes! I agree too...........
  I am giving back control to you.
MESSAGE: (stop, yes, void, void)
Material Agent 1
    Sending Message to Material Critic  ..........

================================================
MATERIAL CRITIC : .............Halted NEGOTIATION.
                  Control Given back to SCHEDULER.
################################################

SCHEDULING AGENT: pc With PRIORITY: 180

Conceptual Design    ############################
          E X P E R T      S Y S T E M
                     F O R
```

```
       P R O C E S S     C R I T I C I S M
################################################
CRITICISM of Process  Selection due to HIGH COST
.............................Starting NEGOTIATION

================================================
PROCESS  CRITIC : NEGOTIATION Started.............
               Control taken away from SCHEDULER.
================================================
PROCESS  CRITIC :  Change the process because cost is high
MESSAGE: (modify, void, cost, high)
Process Critic
    Sending Message to Process Agent 1 ..........

........Negotiation continuing.............
PROCESS AGENT 1:  Though the cost is high, Part has high RPD
MESSAGE: (agree, no, RPD, high)
Process Agent 1
    Sending Message to Process Critic  ..........

........Negotiation continuing.............
PROCESS  CRITIC :  What is your Local Goal ?
MESSAGE: (goal?, local, void, void)
Process Critic
    Sending Message to Process Agent 1 ..........

........Negotiation continuing.............
PROCESS  AGENT 1:  My local goal is to have high RPD
MESSAGE: (goal-is, local, RPD, high)
Process Agent 1
    Sending Message to Process Critic  ..........
```

```
.........Negotiation continuing.............
MATERIAL CRITIC :  I agree to your decision of process choice
    because your goal matches the global goal.
MESSAGE: (agree, yes, void, void)
Process Critic
    Sending Message to Process Agent 1 ..........

.........Negotiation continuing.............
PROCESS  AGENT 1:  Yes! I agree too...........
  I am giving back control to you.
MESSAGE: (stop, yes, void, void)
Process Agent 1
    Sending Message to Process Critic  ..........

================================================
PROCESS  CRITIC : ............Halted NEGOTIATION.
                 Control Given back to SCHEDULER.
################################################

SCHEDULING AGENT: i With PRIORITY: 170

Conceptual Design    ##############################
          E X P E R T      S Y S T E M
                     F O R
       I N S P E C T I O N   A D V I C E
################################################

The inspection method selected is

  Ultrasonic Imaging : C-Scan

Ceramics Applicability
```

```
 GREEN   : Low probability of successful
           application to Ceramics
 SINTERED: Under development for use with Ceramics

Advantages
Fast
Sensitive to Cracks
Disadvantages
Requires a Coupling Agent

SCHEDULING AGENT: mce With PRIORITY: 160

Conceptual Design    ##############################
          E X P E R T     S Y S T E M
                    F O R
M A T E R I A L    C O S T     E S T I M A T I O N
##################################################
The MATERIAL selected was:       ZIRCONIA

     Material Cost   $  0.01 Per Part

           For a  Batch Size of 1000 Parts

SCHEDULING AGENT: pce With PRIORITY: 155

Conceptual Design    ##############################
          E X P E R T     S Y S T E M
                    F O R
P R O C E S S      C O S T     E S T I M A T I O N
##################################################
The PROCESS selected was:     UNIAXIAL DRY PRESSING
```

```
     Process Cost    $  0.91 Per Part

     Total Tooling Cost is   $20000.00

            For a  Batch Size of 1000 Parts

SCHEDULING AGENT: ce With PRIORITY: 140

Conceptual Design    ##############################
          E X P E R T      S Y S T E M
                     F O R
         C O S T     E S T I M A T I O N
##################################################

Based on information from other Cost Estimators

    and for a Batch Size of 1000 Parts

          the Total Cost  is $ 20.92 Per Part


SCHEDULING AGENT: mcc With PRIORITY: 130

Conceptual Design    ##############################
          E X P E R T      S Y S T E M
                     F O R
  M A T E R I A L   C O S T   C R I T I C I S M
##################################################
CRITICISM of Material Cost Estimation due to HIGH COST ESTIMATE
.............................Starting NEGOTIATION

==================================================
```

```
MATERIAL COST CRITIC : NEGOTIATION Started.......
               Control taken away from SCHEDULER.
================================================
MATERIAL COST CRITIC :  Change material! I don't like your estimate!
MESSAGE: (modify, void, cost, high)
Material Cost Critic
    Sending Message to Material Cost Estimator  ......

........Negotiation continuing.............
MATERIAL COST ESTIMATOR:  OK! I will give you a new cost estimate!
MESSAGE: (agree, yes, cost, new)
Material Cost Estimator
    Sending Message to Material Cost Critic     ......

........Negotiation continuing.............
MATERIAL COST CRITIC :  What is your new cost estimate?
MESSAGE: (what-is, void, cost, new)
Material Cost Critic
    Sending Message to Material Cost Estimator  ......

........Negotiation continuing.............
MATERIAL COST ESTIMATOR:  My new cost estimate is $ 0.005 per part
MESSAGE: (void, void, cost, $ 0.005)
Material Cost Estimator
    Sending Message to Material Cost Critic     ......

........Negotiation continuing.............
MATERIAL COST CRITIC :  I agree to your new value of cost estimate!
MESSAGE: (agree, yes, void, void)
Material Cost Critic
    Sending Message to Material Cost Estimator  ......
```

```
.........Negotiation continuing.............
MATERIAL COST ESTIMATOR:  Yes! I agree too...........
  I am giving back control to you.
MESSAGE: (stop, yes, void, void)
Material Cost Estimator
    Sending Message to Material Cost Critic     ......

================================================
MATERIAL COST CRITIC : .......Halted NEGOTIATION.
                Control Given back to SCHEDULER.
################################################

SCHEDULING AGENT: mced With PRIORITY: 30

Detailed   Design    ##############################
           E X P E R T      S Y S T E M
                      F O R
M A T E R I A L    C O S T     E S T I M A T I O N
################################################
The MATERIAL selected was:       ZIRCONIA

     Material Cost   $  0.01 Per Part

            For a  Batch Size of 1000 Parts

SCHEDULING AGENT: pced With PRIORITY: 20

Detailed   Design    ##############################
           E X P E R T      S Y S T E M
                      F O R
P R O C E S S      C O S T     E S T I M A T I O N
################################################
```

```
The PROCESS selected was:     UNIAXIAL DRY PRESSING

     Primary Process Cost    $  0.91 Per Part

     Secondary Process Cost  $  0.00 Per Part

     Tooling Cost            $ 20000.0

            For a  Batch Size of 1000 Parts
Note :
   Secondary Process was not selected.

SCHEDULING AGENT: iced With PRIORITY: 10

Detailed Design   ################################
          E X P E R T      S Y S T E M
                     F O R
I N S P E C T I O N  C O S T   E S T I M A T I O N
################################################
The INSPECTION method selected was:      C Scan

     Inspection Cost         $ 35.00 Per Part

            For a  Batch Size of 1000 Parts

SCHEDULING AGENT: ced With PRIORITY: 5

Detailed Design   ################################
          E X P E R T      S Y S T E M
                     F O R
        C O S T    E S T I M A T I O N
################################################
```

```
CONFLICT for Detailed CE.....................
      Value already decided by Conceptual CE
.............................Starting NEGOTIATION

================================================
DETAILED   CE: NEGOTIATION Started.............
               Control taken away from SCHEDULER.
================================================
DETAILED   CE:  Change the cost value because, I have a precise value
MESSAGE: (modify, void, cost, imprecise)
Detailed CE
    Sending Message to Conceptual CE .............

........Negotiation continuing.............
CONCEPTUAL   CE:  What is your Local Goal ?
MESSAGE: (goal?, local, void, void)
Conceptual CE
    Sending Message to Detailed CE ...............

........Negotiation continuing.............
DETAILED   CE:  My local goal is to precisely estimate cost
MESSAGE: (goal-is, local, cost, precise)
Detailed CE
    Sending Message to Conceptual CE .............

........Negotiation continuing.............
CONCEPTUAL   CE:  I agree to your precise value of cost estimation.
MESSAGE: (stop, yes, void, void)
Conceptual CE
    Sending Message to Detailed CE ...............

================================================
```

```
DETAILED   CE:    ............Halted NEGOTIATION.
                 Control Given back to SCHEDULER.
##################################################
```

Appendix C

Traces from I3D+ Negotiation System: Different Conflict Situations

---

This section presents the execution of the I3D+ system with two different conflict situations implemented in the system. This trace shows how the different conflict situations are handled and resolved in the system.

```
##################################################

   Agents in AGENDA with their priority rating

##################################################
p1 197
p2 195
pc 190
m1 189
m2 185
mc 180
i 170
mce 160
```

```
pce 155
ce 140
mcc 130
mced 30
pced 20
iced 10
ced 5
##################################################

       Entering    SCHEDULING     Loop

##################################################

SCHEDULING AGENT: p1 With PRIORITY: 197

Conceptual Design    ##############################
          E X P E R T      S Y S T E M
                    F O R
    M A N U F A C T U R I N G    P L A N N I N G
     ( P R O C E S S      S E L E C T I O N  1)
##################################################
NO CONFLICT in process selection...
......even though process was already decided!

The Process Selected is

   Uniaxial Dry Pressing

SCHEDULING AGENT: p2 With PRIORITY: 195

Conceptual Design    ##############################
          E X P E R T      S Y S T E M
```

```
                    F O R
    M A N U F A C T U R I N G    P L A N N I N G
     ( P R O C E S S      S E L E C T I O N  2)
##################################################
CONFLICT in process selection...
.................... Starting NEGOTIATION

The Process Selected is

   Uniaxial Hot Pressing

==================================================
PROCESS  AGENT 2:  NEGOTIATION Started...........
               Control taken away from SCHEDULER.
==================================================
PROCESS  AGENT 2:  Change the process because LDR is Low
MESSAGE: (modify, void, LDR, low)
Process Agent 2
    Sending Message to Process Agent 1 ..........

........Negotiation continuing.............
PROCESS AGENT 1:  Though the LDR is low, Part has high RPD
MESSAGE: (agree, no, RPD, high)
Process Agent 1
    Sending Message to Process Agent 2 ..........

........Negotiation continuing.............
PROCESS AGENT 2:  What is your Local Goal ?
MESSAGE: (goal?, local, void, void)
Process Agent 2
    Sending Message to Process Agent 1 ..........
```

```
.........Negotiation continuing.............
PROCESS  AGENT 1:  My local goal is to have high RPD
MESSAGE: (goal-is, local, RPD, high)
Process Agent 1
    Sending Message to Process Agent 2 ..........

.........Negotiation continuing.............
PROCESS  AGENT 2:  I agree to your decision of process  choice
    because your goal matches the global goal.
MESSAGE: (agree, yes, void, void)
Process Agent 2
    Sending Message to Process Agent 1 ..........

.........Negotiation continuing.............
PROCESS  AGENT 1:  Yes! I agree too...........
  I am giving back control to you.
MESSAGE: (stop, yes, void, void)
Process Agent 1
    Sending Message to Process Agent 2 ..........

================================================
PROCESS  AGENT 2:.............Halted NEGOTIATION.
                 Control Given back to SCHEDULER.
################################################

SCHEDULING AGENT: pc With PRIORITY: 190

Conceptual Design    ############################
          E X P E R T      S Y S T E M
                     F O R
       P R O C E S S      C R I T I C I S M
################################################
```

```
CRITICISM of Process  Selection due to HIGH COST
.............................Starting NEGOTIATION

================================================
PROCESS  CRITIC : NEGOTIATION Started.............
               Control taken away from SCHEDULER.
================================================
PROCESS  CRITIC :  Change the process because cost is high
MESSAGE: (modify, void, cost, high)
Process Critic
    Sending Message to Process Agent 1 ..........

........Negotiation continuing.............
PROCESS AGENT 1:  Though the cost is high, Part has high RPD
MESSAGE: (agree, no, RPD, high)
Process Agent 1
    Sending Message to Process Critic  ..........

........Negotiation continuing.............
PROCESS  CRITIC :  What is your Local Goal ?
MESSAGE: (goal?, local, void, void)
Process Critic
    Sending Message to Process Agent 1 ..........

........Negotiation continuing.............
PROCESS  AGENT 1:  My local goal is to have high RPD
MESSAGE: (goal-is, local, RPD, high)
Process Agent 1
    Sending Message to Process Critic  ..........

........Negotiation continuing.............
MATERIAL CRITIC :  I agree to your decision of process choice
```

```
    because your goal matches the global goal.
MESSAGE: (agree, yes, void, void)
Process Critic
    Sending Message to Process Agent 1 ..........

.........Negotiation continuing.............
PROCESS  AGENT 1:  Yes! I agree too...........
  I am giving back control to you.
MESSAGE: (stop, yes, void, void)
Process Agent 1
    Sending Message to Process Critic  ..........

================================================
PROCESS  CRITIC : .............Halted NEGOTIATION.
                 Control Given back to SCHEDULER.
################################################

SCHEDULING AGENT: m1 With PRIORITY: 189

Conceptual Design    ##############################
          E X P E R T      S Y S T E M
                     F O R
    M A N U F A C T U R I N G    P L A N N I N G
      ( M A T E R I A L    S E L E C T I O N  1)
################################################
NO CONFLICT in material selection...
......even though material was already decided!

The material selected is

   Zirconia
```

```
SCHEDULING AGENT: m2 With PRIORITY: 185

Conceptual Design    ##############################
          E X P E R T      S Y S T E M
                     F O R
    M A N U F A C T U R I N G    P L A N N I N G
      ( M A T E R I A L    S E L E C T I O N  2)
##################################################
CONFLICT in material selection...
..................... Starting NEGOTIATION

The material selected is

   Silicon Carbide

==================================================
MATERIAL AGENT 2: NEGOTIATION Started.............
               Control taken away from SCHEDULER.
==================================================
MATERIAL AGENT 2:  Change the material because strength is Low
MESSAGE: (modify, void, strength, low)
Material Agent 2
    Sending Message to Material Agent 1 ..........

........Negotiation continuing.............
MATERIAL AGENT 1:  Though the strength is low, material has high TC!
MESSAGE: (agree, no, tc, high)
Material Agent 1
    Sending Message to Material Agent 2 ..........

........Negotiation continuing.............
MATERIAL AGENT 2:  What is your Local Goal ?
```

```
MESSAGE: (goal?, local, void, void)
Material Agent 2
    Sending Message to Material Agent 1 ..........

.........Negotiation continuing.............
MATERIAL AGENT 1:  My local goal is to increase TC
MESSAGE: (goal-is, local, tc, high)
Material Agent 1
    Sending Message to Material Agent 2 ..........

.........Negotiation continuing.............
MATERIAL AGENT 2:  I agree to your decision of material choice
    because your goal matches the global goal.
MESSAGE: (agree, yes, void, void)
Material Agent 2
    Sending Message to Material Agent 1 ..........

.........Negotiation continuing.............
MATERIAL AGENT 1:  Yes! I agree too...........
  I am giving back control to you.
MESSAGE: (stop, yes, void, void)
Material Agent 1
    Sending Message to Material Agent 2 ..........

================================================
MATERIAL AGENT 2: ............Halted NEGOTIATION.
                  Control Given back to SCHEDULER.
################################################

SCHEDULING AGENT: mc With PRIORITY: 180

Conceptual Design    ##############################
```

```
          E X P E R T      S Y S T E M
                     F O R
        M A T E R I A L    C R I T I C I S M
################################################
CRITICISM of Material Selection due to HIGH COST
.............................Starting NEGOTIATION

================================================
MATERIAL CRITIC : NEGOTIATION Started.............
               Control taken away from SCHEDULER.
================================================
MATERIAL CRITIC :  Change the material because cost is high
MESSAGE: (modify, void, cost, high)
Material Critic
    Sending Message to Material Agent 1 ..........

.........Negotiation continuing.............
MATERIAL AGENT 1:  Though the cost is high, material has high TC!
MESSAGE: (agree, no, tc, high)
Material Agent 1
    Sending Message to Material Critic  ..........

.........Negotiation continuing.............
MATERIAL CRITIC :  What is your Local Goal ?
MESSAGE: (goal?, local, void, void)
Material Critic
    Sending Message to Material Agent 1 ..........

.........Negotiation continuing.............
MATERIAL AGENT 1:  My local goal is to increase TC
MESSAGE: (goal-is, local, tc, high)
Material Agent 1
```

```
    Sending Message to Material Critic  ..........

.........Negotiation continuing.............
MATERIAL CRITIC :  I agree to your decision of material choice
    because your goal matches the global goal.
MESSAGE: (agree, yes, void, void)
Material Critic
    Sending Message to Material Agent 1 ..........

.........Negotiation continuing.............
MATERIAL AGENT 1:  Yes! I agree too...........
  I am giving back control to you.
MESSAGE: (stop, yes, void, void)
Material Agent 1
    Sending Message to Material Critic  ..........

================================================
MATERIAL CRITIC : ............Halted NEGOTIATION.
                 Control Given back to SCHEDULER.
################################################

SCHEDULING AGENT: i With PRIORITY: 170

Conceptual Design    ############################
          E X P E R T      S Y S T E M
                     F O R
       I N S P E C T I O N   A D V I C E
################################################

The inspection method selected is

  Ultrasonic Imaging : C-Scan
```

```
Ceramics Applicability
 GREEN   : Low probability of successful
           application to Ceramics
 SINTERED: Under development for use with Ceramics

Advantages
Fast
Sensitive to Cracks
Disadvantages
Requires a Coupling Agent

SCHEDULING AGENT: mce With PRIORITY: 160

Conceptual Design    ##############################
          E X P E R T     S Y S T E M
                    F O R
M A T E R I A L   C O S T    E S T I M A T I O N
##################################################
The MATERIAL selected was:       ZIRCONIA

     Material Cost   $  0.01 Per Part

           For a  Batch Size of 1000 Parts

SCHEDULING AGENT: pce With PRIORITY: 155

Conceptual Design    ##############################
          E X P E R T     S Y S T E M
                    F O R
P R O C E S S     C O S T    E S T I M A T I O N
##################################################
```

```
The PROCESS selected was:     UNIAXIAL DRY PRESSING

     Process Cost    $  0.91 Per Part

     Total Tooling Cost is   $20000.00

            For a  Batch Size of 1000 Parts

SCHEDULING AGENT: ce With PRIORITY: 140

Conceptual Design    ############################
          E X P E R T      S Y S T E M
                     F O R
        C O S T     E S T I M A T I O N
##################################################

Based on information from other Cost Estimators

    and for a Batch Size of 1000 Parts

          the Total Cost  is $ 20.92 Per Part


SCHEDULING AGENT: mcc With PRIORITY: 130

Conceptual Design    ############################
          E X P E R T      S Y S T E M
                     F O R
  M A T E R I A L   C O S T   C R I T I C I S M
##################################################
CRITICISM of Material Cost Estimation due to HIGH COST ESTIMATE
.............................Starting NEGOTIATION
```

```
================================================
MATERIAL COST CRITIC : NEGOTIATION Started.......
               Control taken away from SCHEDULER.
================================================
MATERIAL COST CRITIC :  Change material! I don't like your estimate!
MESSAGE: (modify, void, cost, high)
Material Cost Critic
    Sending Message to Material Cost Estimator  ......

.........Negotiation continuing.............
MATERIAL COST ESTIMATOR:  OK! I will give you a new cost estimate!
MESSAGE: (agree, yes, cost, new)
Material Cost Estimator
    Sending Message to Material Cost Critic     ......

.........Negotiation continuing.............
MATERIAL COST CRITIC :  What is your new cost estimate?
MESSAGE: (what-is, void, cost, new)
Material Cost Critic
    Sending Message to Material Cost Estimator  ......

.........Negotiation continuing.............
MATERIAL COST ESTIMATOR:  My new cost estimate is $ 0.005 per part
MESSAGE: (void, void, cost, $ 0.005)
Material Cost Estimator
    Sending Message to Material Cost Critic     ......

.........Negotiation continuing.............
MATERIAL COST CRITIC :  I agree to your new value of cost estimate!
MESSAGE: (agree, yes, void, void)
Material Cost Critic
```

```
    Sending Message to Material Cost Estimator  ......

.........Negotiation continuing.............
MATERIAL COST ESTIMATOR:  Yes! I agree too...........
  I am giving back control to you.
MESSAGE: (stop, yes, void, void)
Material Cost Estimator
    Sending Message to Material Cost Critic     ......

================================================
MATERIAL COST CRITIC : .......Halted NEGOTIATION.
                Control Given back to SCHEDULER.
################################################

SCHEDULING AGENT: mced With PRIORITY: 30

Detailed   Design    ##############################
           E X P E R T      S Y S T E M
                      F O R
M A T E R I A L    C O S T     E S T I M A T I O N
################################################
The MATERIAL selected was:       ZIRCONIA

     Material Cost   $  0.01 Per Part

            For a  Batch Size of 1000 Parts

SCHEDULING AGENT: pced With PRIORITY: 20

Detailed   Design    ##############################
           E X P E R T      S Y S T E M
                      F O R
```

```
P R O C E S S     C O S T    E S T I M A T I O N
################################################
The PROCESS selected was:     UNIAXIAL DRY PRESSING

     Primary Process Cost    $  0.91 Per Part

     Secondary Process Cost  $  0.00 Per Part

     Tooling Cost            $ 20000.0

            For a  Batch Size of 1000 Parts
Note :
   Secondary Process was not selected.

SCHEDULING AGENT: iced With PRIORITY: 10

Detailed Design   ################################
          E X P E R T     S Y S T E M
                    F O R
I N S P E C T I O N  C O S T   E S T I M A T I O N
################################################
The INSPECTION method selected was:      C Scan

     Inspection Cost         $ 35.00 Per Part

            For a  Batch Size of 1000 Parts

SCHEDULING AGENT: ced With PRIORITY: 5

Detailed Design   ################################
          E X P E R T     S Y S T E M
                    F O R
```

```
        C O S T     E S T I M A T I O N
##################################################

Based on information from other Cost Estimators

    and for a Batch Size of 1000 Parts

           the Total Cost  is $ 55.92 Per Part


CONFLICT for Detailed CE.....................
      Value already decided by Conceptual CE
.............................Starting NEGOTIATION

================================================
DETAILED   CE: NEGOTIATION Started............
               Control taken away from SCHEDULER.
================================================
DETAILED   CE:  Change the cost value because, I have a precise value
MESSAGE: (modify, void, cost, imprecise)
Detailed CE
    Sending Message to Conceptual CE .............

........Negotiation continuing.............
CONCEPTUAL   CE:  What is your Local Goal ?
MESSAGE: (goal?, local, void, void)
Conceptual CE
    Sending Message to Detailed CE ..............

........Negotiation continuing.............
DETAILED   CE:  My local goal is to precisely estimate cost
MESSAGE: (goal-is, local, cost, precise)
```

```
Detailed CE
    Sending Message to Conceptual CE ............

.........Negotiation continuing.............
CONCEPTUAL   CE:  I agree to your precise value of cost estimation.
MESSAGE: (stop, yes, void, void)
Conceptual CE
    Sending Message to Detailed CE ..............

================================================
DETAILED   CE:    ............Halted NEGOTIATION.
                 Control Given back to SCHEDULER.
################################################
```