CONCURRENT ENGINEERING: Research and Applications

An Agent-Based Decision Network for Concurrent Engineering Design

Mohammad Reza Danesh and Yan Jin*

The IMPACT Laboratory, DRB 101, University of Southern California, Los Angeles, CA 90089-1111 USA

Received 20 October 2000; accepted in revised form 07 January 2001

Abstract: Engineering design is a complex process. Even in designing a simple product, many decisions must be made by a designer. The problem gets more complex when multiple designers work as a team to design single artifact. In this paper, we take a *decision-based* approach to model design process and introduce an agent-based decision network (ADN) to support concurrent decision-making and collaboration in design. ADN focuses on making designers consider other team members' decisions when making their own and attempts to achieve coherent design decisions among designers by explicitly representing and enhancing individual design decision-making and negotiation processes. ADN is composed of a decision-based design process model (DDPM), an objective-based negotiation model (OBNM), and a number of intelligent agents, each associated with a human designer. The DDPM was developed to capture individual designers' design processes. OBNM was developed to facilitate objective-based negotiation and to track both dependencies generated and decisions made at each design stage for downstream negotiation support. In ADN, each designer is associated with an agent and both the DDPM and OBNM are captured and facilitated by agents and are not explicitly visible to designers. Agents generate and utilize the DDPM and OBNM information to support their designers. This paper describes the ADN framework in detail, points out its advantages, and presents an application example to demonstrate the effectiveness of the ADN framework.

Key Words: design process, group decision-making, design values, coordination, design dependencies, intelligent agents, concurrent engineering.

1. Introduction

Engineering design is a complex process. Even in designing a simple product such as a gear, many decisions must be made by a designer. The problem gets more complex when a group of designers work as a team to design single artifact. With the increasing competition in time to market and shortened design cycle time, concurrent execution of design tasks is needed, and design teams must exchange information with each other, find and resolve design conflicts, generate new ideas and design alternatives, and assess the manufacturability of at early stage of design. As the complexity of designed artifact increases, maintaining effective and efficient concurrency becomes a major challenge in team design.

Researchers in the area of engineering design have done extensive study on design methodologies. Some examples are Axiomatic Design Model [19], Systematic Design Model [14], and Decision-Based Design model [3,4,12,13]. Axiomatic Design Model identifies two axioms to be fundamental (namely independence axiom and information axiom). The former suggests maintaining independence between functional requirements and the latter suggests minimizing the information content. Systematic design model is based on the observation that engineering design must be carefully planned and systematically executed and that a design method must integrate many different aspects of engineering. Some of the decision-based design models [3,5] take a decision theoretic approach to engineering design by applying classical decision theories [22], and others [12,13] try to define design problems in algebraic forms and solve them with optimization algorithms.

As much as local design is important and essential in team design process, the key action that differentiates concurrent design from isolated design is coordination. Coordination is needed to manage interdependencies between designers and to facilitate progress of each individual [10,17]. Researchers have done substantial research in fields of concurrent engineering, computer support for collaboration work (CSCW) and distributed artificial intelligence. Some examples are action workflow model [11], contract-net model [18], Redux model [16], distributed problem solving through coordination (CP&CR) model [9], and generalized partial global planning (GPGP) model [2].

The action workflow model has been developed in a series of systems for coordination among users of networked computers. This model characterizes a workflow based on the identification and construction of atomic "loops" of action in which a performer completes an action to the satisfaction of a customer. Contract-net model emphasizes the issues of task distribution in a distributed problem-solving environment. Redux model, a centralized decision maintenance server, keeps track of decisions made by the team members and maintains the dependencies between them. CP&CR (Con-

*Author to whom correspondence should be addressed. E-mail: yjin@usc.edu

1531-2003/01/01 0037–11 $10.00/0 DOI: 10.1106/P3P6-3HEA-GXB2-9MTT

straint Partition and Coordination Reaction) model is a problem-solving framework in which a society of specialized and well-coordinated agents collectively solves a problem. GPGP model consists of an extensible set of modular coordination mechanisms. Each mechanism is defined as a response to a certain feature in the current subjective task environment. GPGP works in conjunction with an agent architecture and local scheduler. GPGP approach views coordination as modulating local control but doesn't replace it.

Some literature on concurrent engineering [15] emphasizes bringing the downstream design process to upstream to avoid conflicts with other designers' actions. However, in implementation, the interaction between activities becomes extremely high when local design processes have to consistently exchange information with team requirements and resources. In the existing literature these interactions are referred to as interactions between tasks, talents, tools, time, technology, techniques, and teamwork (7Ts). Coordination models can give members of a team (whether they are designers or manufacturers) enough support to overcome such inconsistencies by using negotiation models. In practice however, design processes often become inconsistent with each other. A small change in one local design decision can create several inconsistencies in the overall design and may require several rounds of conflict resolution among members of the design group [1,7]. In such cases, the challenge for CE models is to provide support to avoid, detect or resolve the conflicts at the stage that they are created, reduce further coordination, and increase efficiency of design process.

This paper is an attempt to develop a framework to support concurrent design decision-making and group coordination by introducing an integrated process model for design decision-making and negotiation. Our objectives are 1) to develop a decision-based model to capture local design decision making, 2) to develop a negotiation model to support coordination and resolve conflicts among team members based on the local design process, and 3) to develop an agent-based system to support concurrent design decision making. The basic idea is that group design process has two aspects: First is local design process where a single designer solves his/her own design problem. Second is coordination between members on their design plans and solutions in order to reach coherent local decisions by considering each others' decisions. In order to increase the effectiveness and efficiency of the concurrent design process, these two steps should be executed consistently. Hence, the focus of this paper is on how to develop a framework and mechanisms to support local decision-making and team coordination dynamically during the process of design.

In the following sections, we first introduce and discuss the idea of agent-based design decision network for collaborative engineering design. After that, we present model of design decision-making and coordination. We then describe our agent-based implementation of the models and present an example application. Finally, we summarize this paper and point out future research directions.

2. Design Decision Network for Collaborative Design

Large scale and complex design problems require engineers to be able to apply principles and methodologies of design and collaboration, to efficiently use emergent computer technologies, and to work effectively in teams. The existing support methodologies for concurrent engineering are anchored around the idea of providing extensive amount of downstream information to upstream decision points so that designers can make better decisions. The most prevalent method to achieve this objective is based on the idea of "data sharing." The problem with this approach is that designers often have trouble using the huge amount of information they receive effectively, and in most cases are overwhelmed with information. The second view can be characterized as "group decision support (GDS)." Here the focus moves away from information to the process of group decision-making, which traditionally takes the form of group meetings. Members of the product development team arrange meeting from the early stages of the process, and try to forecast and avoid possible inconsistencies and conflicts.

While group decision-making is one aspect of collaborative design, the existing GDS framework does not explicitly address concurrent decision making processes and their various linkages. As a result, most GDS tools fail to provide engineering design specific solutions and support other than facilitating mail exchange and telephone or video conferencing.

To overcome the shortcomings of the existing frameworks for design process, we propose an "Agent-based decision network (ADN)" framework. Our principal claim is that collaborative design is not merely about data. It is about the processes of decision making that are carried out by multiple designers in specific organizational (functional and social) contexts and involves applications of specific design knowledge. The ADN view is different from GDS in that instead of focusing only on group meetings, the ADN thinking emphasizes the roles of individuals' decision processes and the links between those processes. This distinction is crucial for concurrent engineering because the key contents of concurrent engineering are individual designers' design processes and their coherent links. Group meeting is only a "snapshot" of the whole process and may not provide a complete understanding of the concurrent design process.

The ADN view of concurrent engineering design recognizes two key actions each performed or exercised at different levels. These actions are 1) decision-making by individual designers using decision-based design process model, and 2) coordination between designers on dependent activities.

Design decision-making: Design starts from the need to find a solution for a design requirement (e.g. to design mechanical components for a car headlight). The process of moving from the initial requirement to the final solution could involve hundreds, or thousands decisions. We are con-

cerned with decision-making aspect of design because decisions of a designer have to be coordinated with others. Inspired by Petrie [16], we view the local design decision-making process composed of a set of tasks, decisions, and solutions, as shown in Figure 1.

A task can be decomposed into sub tasks, or single solution of the task can be found, by a making a decision. A *design task* represents a requirement, which could be functional or physical. Designers arrive at and make their decisions, by applying a systematic process [8] of defining design objective, generating design alternatives, evaluating the alternatives, and selecting the best alternative. To develop a framework to support concurrent decision-making, we must explicate knowledge for alternative generation, formalize procedures for choice analysis and selection, and develop a framework for decision-making dependency management. The focus of this paper is on developing representation constructs of the various elements of the decision-making process.

Coordination: Design coordination is defined as interactive actions between designers. Although designers try to make sub-tasks independent, they have to coordinate the outcomes of their decisions with downstream designer or manufacturers. These coordination activities in different design stages are for different purposes (conflict resolution, constraint satisfaction, dependency identification, etc). In concurrent engineering design, dependencies are the links that bridge individual design decision processes to each other and create the design dependency network as illustrated in Figure 2. This figure illustrates a design scenario where three designers (A, B, and C) are collaborating on a design problem as a design team. Each designer is responsible for several design tasks; i.e., s/he has control of generating alternatives, realizing the consequences, and evaluating the alternative with respect to the requirements of these tasks. We define the union of these tasks as responsibility boundary of that designer. Thick lines between design tasks represent the dependencies among them. We refer to the complete graph of all design tasks with the links as *design topology*. Design topology graph can be used to illustrate how tasks are dependent on each other and to find solutions for tasks, which dependencies should be considered and satisfied. In real design practice, obtaining the design topology graph at the early stages of design process is difficult.

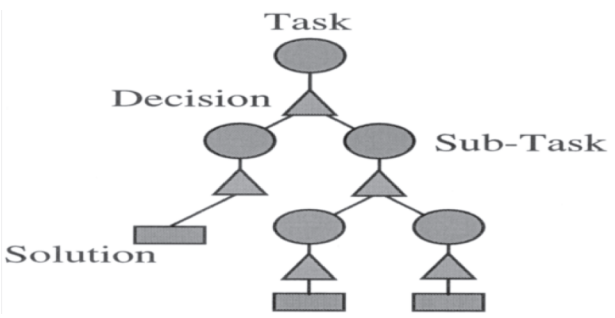

Figure 1. Design decision-making process.

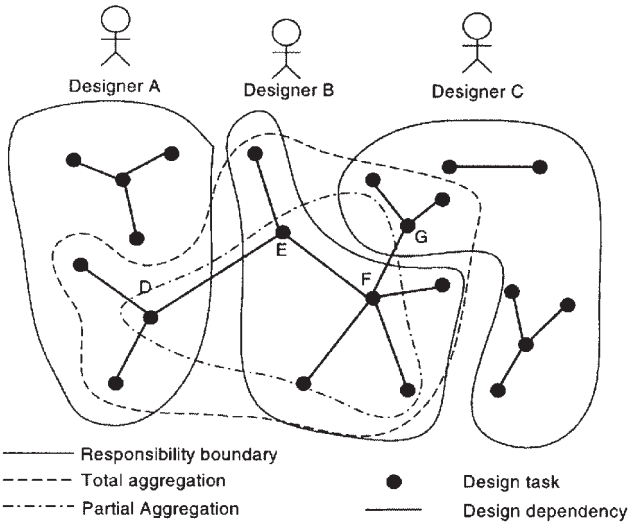

Figure 2. Distributed, partial, and total aggregation of design tasks.

In order to achieve efficient coordination, designers have to go beyond coordinating design activities and coordinate their *design values* through considering and aligning each others' design objectives. Design value is what a designer ultimately cares about in a decision situation. Based on this idea, we introduce the concept of *value level coordination.* The basic idea of value level coordination is that designers' design values determine how design will proceed and how design results will be generated. If designers can coordinate and agree on their values first, they will most likely proceed with their designs in compatible ways, and consequently their design results will be compatible. As a result, adequate value level coordination can significantly reduce the need for task level or result level coordination. Design values are generated by designers through a systematic decision making process, based on the design objectives of the current decision problem. Hence, our focus is on aggregating design values in order to achieve effective collaboration. We recognize three levels for value aggregation: zero aggregation, partial aggregation and complete aggregation as illustrated in Figure 2.

Zero aggregation refers to situations where multiple designers carry out design tasks among themselves without sharing their design values. Zero aggregation of design values is similar to situations where designers work separately and don't consider the impacts of their decisions on other team members. In such cases, conflicts are only recognized after decisions are made. Hence, designers have to backtrack from their decision and reevaluate other design alternatives in order to resolve conflicts. With zero aggregation, designers are sharing their activities at the result level. Partial aggregation, which is represented by centerlines in Figure 2, refers to situations where design values are shared between parts of dependent design tasks. These activities might not necessarily belong to one designer's responsibility boundary. We use the term partial because it applies to situations where design

dependencies between design tasks are not completely established (either because design process is not completed or designers don't recognize the dependency). Complete aggregation represented by dash lines in the same figure is a case where design dependencies among design tasks are completely identified. In both partial and total aggregation, designers move away from result level sharing to value level sharing which is desirable, since value level sharing can not only increase the overall design values, but it also reduces the need for result level coordination, and consequently reduce the total need for coordination.

To provide support for effective coordination, we must develop protocols for coordination and identify knowledge that can be applied to coordinate decisions. As discussed in the next section, we have developed an objective-based negotiation protocol (OBNP) to support coordination of design activities.

The important feature of our ADN framework is that it explicitly captures decisions and coordination links in the sense that it provides a structured representation of the concepts and actions as well as the mechanisms to support the actions. It also facilitates objective sharing among dependent design activities and provides further insights for designers in generating alternatives compatible with other designers' values. The main challenge we face to realize the framework is: how can we develop effective models for agents to capture and support decision-making, objective sharing, and coordination? The following section presents our current model.

3. Modeling Design Decision Making and Negotiation

We aim at developing a formal design process model to provide unambiguous representation of the required information and operations so that the mechanisms for supporting those can be defined. We introduce decision-based design process model to serve the following needs: 1) to formalize design process information so that it can be recorded for later usage; 2) to define a design process information structure that can be used to identify and deal with dependencies between designers' objectives and tasks; and 3) to provide needed concepts and basis for composing a negotiation protocol. In the remaining sections, we begin by presenting main concepts of the decision-based design process model (DDPM), and objective-based negotiation model (OBNM). After that, we show how these two models are combined together in the ADN framework.

4. Elements of DDPM

In a very general term, design is a process of generating information. To capture the design process, we should record the related and useful information. We recognize that design tasks, alternatives, decisions, and solutions represent the key concepts of a design process. We define design process as follows:

4.1 Definition 1: Design Process

A *design process* denoted by $DP(t)$ is defined as:

$$DP(t) = \{T, Alt, D, S\}$$

where

$T = \{t_1^0, t_2^1, t_3^1, t_4^2, t_5^2 \ldots t_n^m\}$: Tasks

$Alt = \{alt^1, alt^2, \ldots alt^k\}$:

Alternative set $AS^i = \{Alt_1^i, Alt_2^i, Alt_3^i \ldots\}$

$D = \{d_1^1, d_2^1, d_3^2, d_4^2 \ldots\}$: Decisions,

$O = \{o_1^1, o_2^1, o_3^2, o_4^2\}$: Objectives,

$S = \{s_1, s_2, s_3 \ldots\}$: Solutions.

t_1^0 represents the initial task.
(This task is not decomposed from any other task.)

4.2 Definition 2: Tasks

A *Task*, denoted by t, is associated with a set of objectives O and attributes A and is defined as

$$t_n^m = \{O, A\} = \{o_1, \ldots, o_p, a_1, \ldots, a_q\}$$

where , $o_i \in O$, and $a_i \in A$, and m is the task number of the original task from which the task is generated, and n is a unique identification number for task.

Task objective is the objective function of a task. *Task attributes* are measures to verify which alternative meets the requirements of task objectives. In order for an attribute to be useful in decision-making process, it should be comprehensive and measurable. *Comprehensive* attribute is an attribute, which by knowing the level of it in a particular situation, there is a clear understanding of the extent that the associated objectives can be achieved. *Measurable* attribute is an attribute, which a) is reasonable to obtain a probability distribution for each alternative over the possible levels of the attribute and b) is reasonable to assess the designer's preferences for different possible levels of the attribute [7].

4.3 Definition 3: Alternatives

An *Alternative* is a set of possible sub-tasks and/or solutions, which can satisfy the objective(s) of a certain task. Each alternative set, AS^k, represents one possible course of action to satisfy a design task. The complete set of alternative sets creates an *alternative space.*

$$AS = \{alt^1, alt^2, \ldots, alt^k\} \equiv \{[alt_1^1, alt_2^1, alt_3^1 \ldots],$$

$$[alt_1^2, alt_2^2, alt_3^2 \ldots][alt_1^k, alt_2^k, alt_3^k \ldots]\}$$

Alternative generation is a creative process of human designers. Our on-going research attempts to develop better understanding of this process. Generally, alternatives in ADN can be generated using one of the following three methods:

1. Based on designer experience (creative alternatives). Designers use their knowledge to generate required steps to satisfy the design objective(s).
2. Alternatives can be selected from a *Pre-recorded* set of alternatives. Designers use library of previously recorded alternatives to find the proper alternative, which suits the assigned task. A suitable method for such cases is to use group technology in order to find similar design cases.
3. Designers can generate alternatives by using alternative generation tools. Software tools can assist the designer in generating solutions. We are currently developing models and tools to assist alternative generation.

4.4 Definition 4: Design Decision

A *Design Decision*, denoted by "d" links a task t_n^m to a set of subtasks $\{t_1^n, t_2^n \ldots\}$ or a single solution s_n found in the alternative space O.

The process of decision-making has three steps, 1) generation of design objectives, 2) evaluation of design alternatives, and 3) selection of one alternative. In our DDPM model, we represent decisions as follows: d_j^i.

$$D = \{d_1^1, d_2^1, d_3^2, d_4^2 \ldots\}$$

"i" represents the task for which decision is made and "j" represents the alternative to which the decision leads.

4.5 Definition 5: Solution

A *solution*, denoted by s, is the result of a decision made on an alternative set AS of task t.

Solutions can be either decomposable or non-decomposable depending on the extent of the knowledge of the designer. If further decomposition of a solution is possible, it's an *intermediate solution* and it becomes a new set of tasks for the next level of decomposition. If it's non-decomposable, it's a *final solution* and no further decomposition is possible.

Some examples of solutions are catalogue references, physical dimensions, product models, and shifting design responsibility to another designer.

5. Objective-Based Negotiation Model (OBNM)

The DDPM model so far captures the local decision-based design process. In order to cope with the collaboration aspects of design, local design processes of individual designers should be coordinated coherently and consistently. In previous sections, we argued that designers have to consider other designers' values and objectives to achieve consistent overall design outcomes. Hence, a coordination model is required to facilitate the value sharing among design teams. "*Objective-based negotiation model (OBNM)*" was developed to cope with this requirement.

Negotiation is the process of interaction between two (or more) parties to reach a mutual agreement to resolve conflicts and/or dependencies. Specifically in team design context, negotiation is required between designers to find out whether the outcome of a design task affects other design tasks. Consider the following example from car manufacturing industry. A harness-wiring designer needs 3 cm^2 space between engine and body frame to pass the headlight cables. If this designer is responsible of the whole system, s/he could change other requirements (based on the priority of this requirement) to accommodate this need. However, in most cases, another designer is responsible for structural design and components arrangements, and the wiring designer can't change parameters of other designers' parts without prior coordinating with the structural designer. Such negotiations are potentially time consuming and require structural designer to realize the harness designer's values in order to generate a suitable common alternative.

In OBNM, coordination activities are classified into two types: 1) Value level coordination, 2) Result level coordination. For each type, several coordination actions are possible. The coordination primitives corresponding to each class are identified as shown in Figure 3. The *collection* of all coordination primitives presents the "*coordination taxonomy*."

As illustrated in Figure 3, OBNM recognizes two families of coordination actions, namely value level and result level. Value level coordination actions are required for negotiating the design objectives, task proposals, and task assignments. In this type of negotiation, designers coordinate their "approach" to design process and convey their design rationale to other team members. It will help other designers to understand local designer's values and objectives. Consequently, another member will work with local designer to reach a common solution. Result level coordination actions are required for negotiating design results. It refers to coordination actions that contribute to consistency of design outcome, for example, the allocated space between engine and frame for harness wiring assigned by the structural designer.

In order to further illustrate the coordination actions, the three cases are selected from the value-level coordination. Assuming communication protocols are already established, consider a design team Tm consisting of n designers is collaborating on finding a design solution for a particular design task. Initially (at time $T = t$), each designer has a set of Tasks T_j, Design Objectives O^j, and Alternatives AS^j.

1. Propose design task: At time $T = t'$, designer 1 (D^1) proposes the design task T_i^1 to the dependent designer(s) or

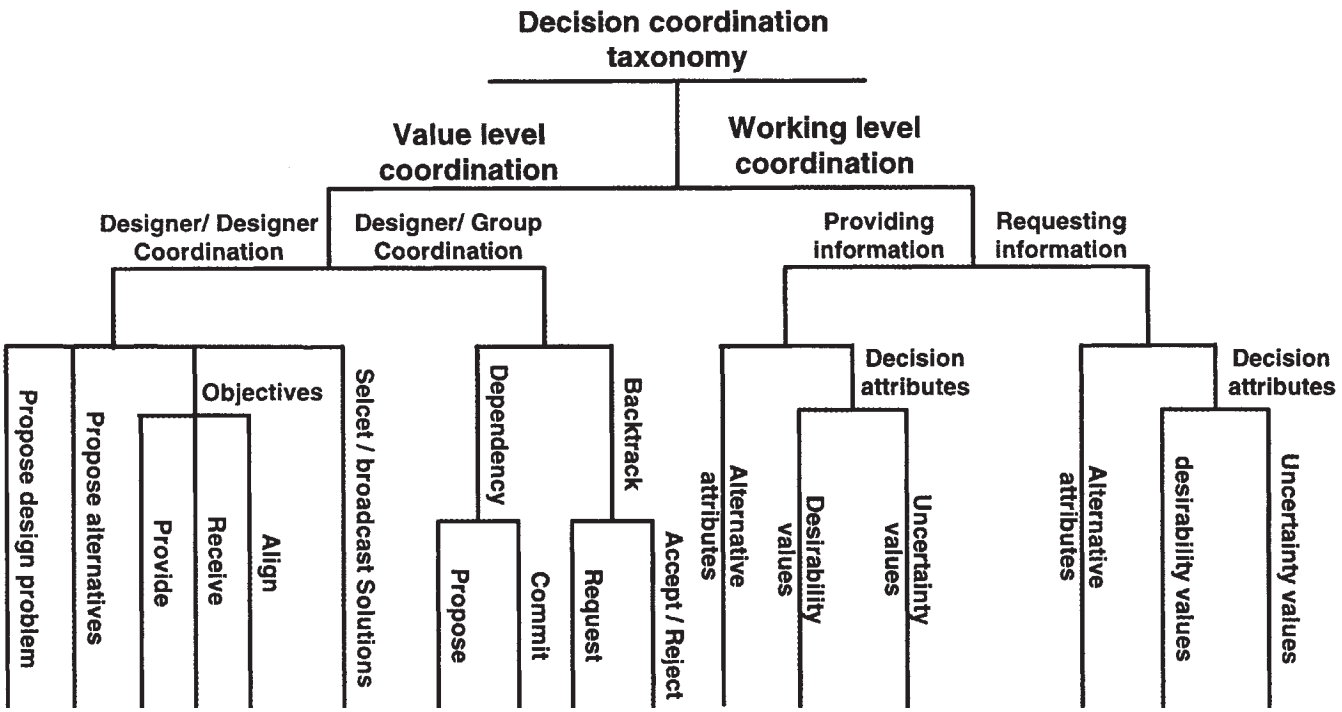

Figure 3. OBNM coordination taxonomy.

broadcasts the task to the team to find a dependent designer. Dependent designer (D^2) checks if the proposed task is consistent with local knowledge. If so, it accepts this dependency and adds it to its task list:

$$T_{2,j}^{t'} = T_{2,j} \cup T_i^1 \text{ if and is consistent.}$$

If the proposed task is inconsistent with local tasks, D^2 rejects the proposal and possibly another round of negotiation is required (Figure 4).

2. Propose design alternative: After receiving the acceptance of task from designer D^2, designer D^1 provides its objectives and alternatives to designer D^2 (Figure 4).

$$\forall assertion\ (AS_1^{j,t'}) \therefore AS_2^j \cup AS_1^j, \text{ and}$$

$$\forall assertion\ (O_1^j) \therefore O_2^{j,t'} = O_2^j \cup O_1^j$$

3. Objective alignments: Happens where design objectives of dependent designer are inconsistent with objectives of proposing designer (Figure 5).

$$\text{if } O_2^j \text{ is inconsistent with } O_1^j \Rightarrow \text{retrack } O_2^j$$

$$\text{request new}(O_2^j)$$

$$\forall\ assertion(O_2^j) \Rightarrow O_1^{j,t'} = O_1^j \cup O_2^j$$

In ADN, OBNM is used in conjunction with DDPM. At each stage, during the evaluation of an alternative in DDPM, if the designer finds a conflict or need of negotiation with another member, s/he uses OBNM as a framework for communication. Depending on the nature of the design problem, s/he can control, exchange information or negotiate with other designers.

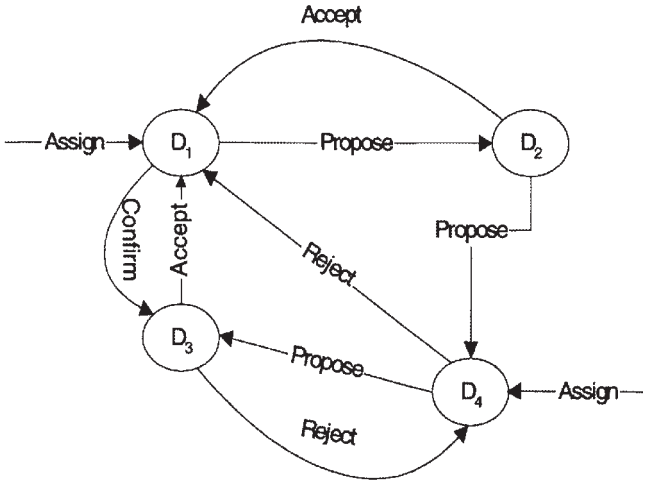

Figure 4. Task proposal state diagram.

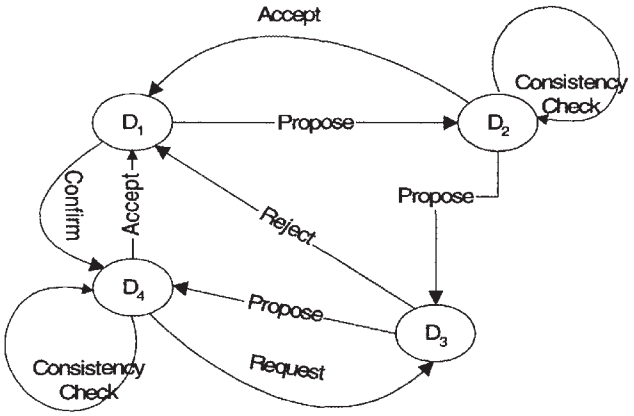

Figure 5. Objective alignment state diagram.

6. Agent-Based Implementation of Design Decision Network

In this section, we will present our implementation of ADN framework. Current research on agent-based support tools for concurrent engineering are mostly concerned about consistently issues of concurrent product development process [20, 21, 23], although some recent works tried to address the issues on dynamics of task distribution [7]. In our research, we adopted the agent-based approach to provide support for distributed decision-making process. Agents act as supporting tools to maintain the local design values and their dependencies with other team members' values. Our implementation of an agent-based system uses JAVA[1] language. In the following sections, we present a detailed view of the ADN agent architecture and an example case that was used to verify the implementation.

7. ADN Architecture

In ADN framework, each design station consists of a designer, a CAD system, and an agent associated with the CAD system as illustrated in Figure 6. Designers interact with both local CAD system and their associated agent. Agents communicate with both local CAD system and other agents using KQML message passing language.

Each agent has six major components. They are: user-interface module, design process-capturing module, rule based engine and conflict detection module, negotiation module, alternative space storage module, and finally the communication module.

User-interface module is in charge of all the interfaces with the designer. User will interact with the user interface in each step of design process, receive results of negotiations with other designers, and store the design decisions to the agent.

Design process-capturing module is in charge of capturing the DDPM steps. Throughout the design process at each

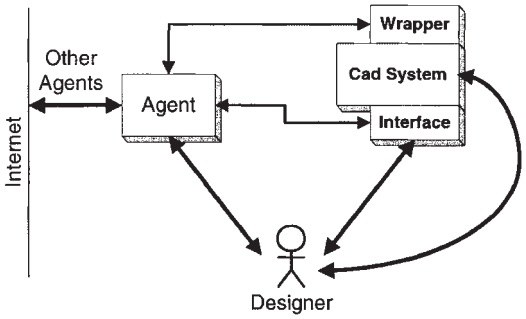

Figure 6. Interactions between designer, ADN agent, and the CAD system.

decision point, designer may generate one or more alternatives. The alternative space storage module is in charge of storing these alternatives. Once an alternative is selected, the process of evaluation and negotiation will be started. The decision making process for this task will be finished when evaluation process is done and one of the qualified alternatives is selected. Design process-capturing module has a viewer tool to view the recent decisions. It also maintains the information about dependency links between local tasks and global design process in case there is a change in the selected alternative in future (Figure 7).

Alternative space storage module is in charge of storing the alternatives for each local design decision. In our current implementation, the alternatives are generated by the designer and are captured by the system, but one can easily replace this implementation with an alternative generation tool or an agent, which stores pre-recorded alternatives (Figure 8).

Rule based engine and conflict detection module is in charge of finding the possible conflicts amongst the selected alternatives. This module will act whenever designer evaluates a new alternative or when another designer in the team proposes a new task. In our implementation we used JESS©[2] inference engine for this purpose. JESS is an implementation of CLIPS©[3] expert system for Java language. The rule-based files for each designer have two parts (i.e. general rules and discipline-based rules).

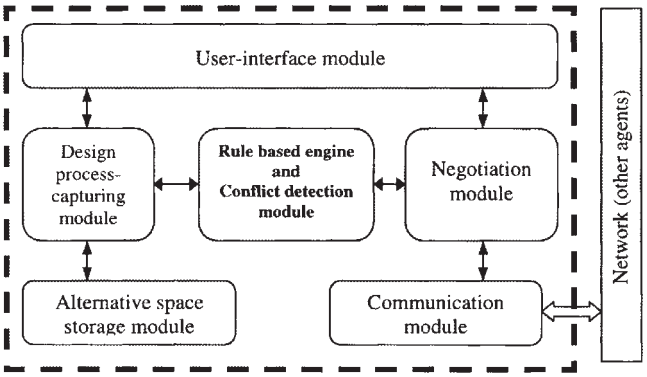

Figure 7. ADN agent architecture.

[1] http://www.javasoft.com

[2] http://herzberg.ca.sandia.gov/jess/

[3] http://www.ghg.net/clips/CLIPS.html

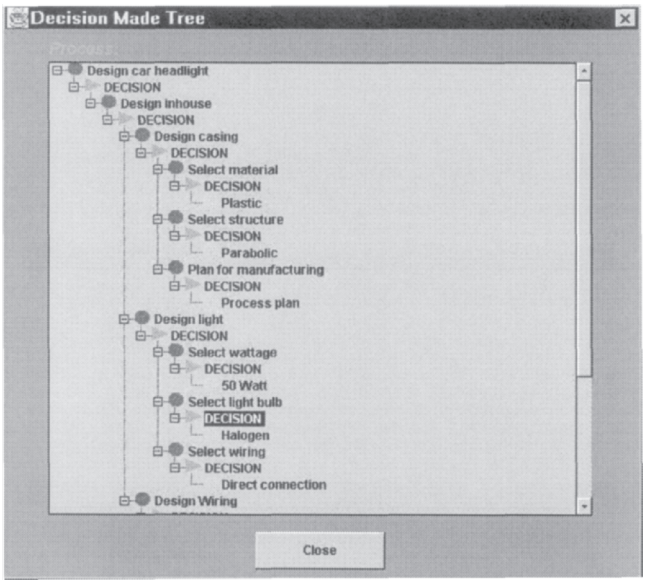

Figure 8. Design process capturing window for mechanical designer.

Negotiation module is in charge of initiating negotiation with other designers. Once a conflict is detected by the rule based engine and is reported to the negotiation module, this module will find the corresponding designer and create an appropriate message. Depending on type of conflict reported by the inference engine, appropriate message is generated. If there is a change in the local design process, the module will create a proposal message for the corresponding designer(s). Proposals received from another agent will be delivered to the inference engine for evaluation.

Communication module is in charge of connecting to the network, sending, and receiving messages. We used "*knowledge query and manipulation language (KQML)*"[4] as language for our agent communications. The communication module converts the messages sent by the negotiation module to a KQML message and sends it to the proper agent. At the same time, if the communication module receives a message from another agent, it will extract the information and report it to the negotiation module for interpretation.

The above-described architecture is a general architecture for the ADN model agent that was implemented for this research. In the following section, we will present a case study that was developed in order to verify the proposed conceptual framework and to test its performance in a design team.

8. Case Example

In order to check the validity of ADN model, we developed a case study to understand the feasibility and implications of our model in solving real world problems. In this example, our objective is to design and manufacture a car headlight for a new model of a car. The project manager in charge of the front section design lays out the basic requirements and constraints to the design team consisting of four members (i.e., mechanical designer, electrical designer, style designer, and manufacturing engineer). The top-level task for each designer is to design the corresponding components of the headlight and for the manufacturing engineer is to take the product models (CAD drawings) and design the appropriate process plan for the designed components. To illustrate the applications of the ADN framework, let us concentrate on the manufacturing engineer for the time being. One of the major problems in traditional design processes is that manufacturers are not involved in the design until final stages of design and hence the manufacturability issues are not considered.

Task 1: Decide on structure: The mechanical designer has to make a decision on the structure of the headlight as a task assigned to him/her (Figure 9). Two alternatives that are available to the designer are 1) parabolic and 2) spherical geometry.

$$AS_{ME} = \{O^1, O^2\},$$

$$O^1 = \{Parabolic\ geometry\},$$

$$O^2 = \{Spherical\ geometry\}$$

The mechanical engineer's design decision is closely dependent on the manufacturing engineer's decisions. Since the spherical geometry is most common geometry in designing a car headlight, let us assume that the mechanical designer starts his evaluation with this alternative. His corresponding agent sends a message to the dependent designer (manufacturing in this case) informing him about this selection. The manufacturing engineer's agent receives the information and adds it to its knowledge.

$$AS^{t'}_{MFE} = AS^{t}_{MEF} \cup O^{t}_{1,ME}$$

Manufacturing engineer may agree to the proposal since the manufacturing facilities are capable to manufacture such geometry. In this case, the proposal will be asserted to the existing alternative set and the corresponding agent sends back the agreement to the proposing agent. No further negotiation is required.

Task 2: Decide on material: in order to make decision on the material type for the casing, mechanical designer should generate alternatives. In this case, designer generates two alternatives: 1) plastic, 2) glass.

The material selection has a significant impact on the manufacturing process. So there is a dependency between this task's outcome and manufacturing engineer's preferences. Now suppose the mechanical designer starts his design task by evaluating the "glass" alternative for the headlight. S/he uses his/her agent to propose this alternative to the manufacturing engineer.

[4] http://www.cs.umbc.edu/kqml/

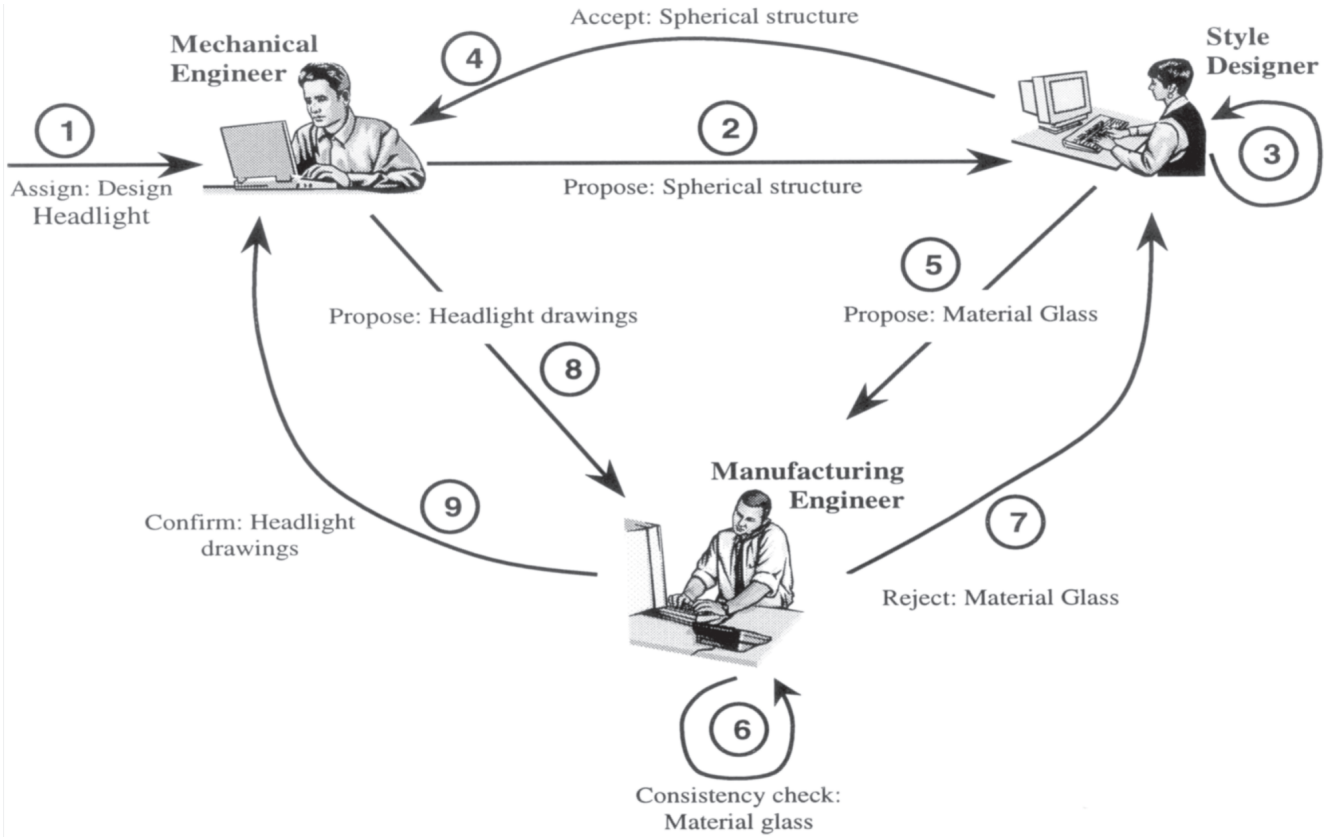

Figure 9. Example of the negotiation process in ADN (car headlight design).

$$AS_{ME} = \{O_3^1, O_3^2\}$$

$$O_3^1 = \{Plastic\}, O_3^2 = \{Glass\}$$

The manufacturing engineer receives his/her proposal, evaluates it, and rejects this selection because the manufacturing department isn't equipped with machinery to build the headlights from glass with the required constraints.

$$Assert(AS_{MFE,3}^{t'}) = AS_{MFE,3}^{t} \cup O_{ME,3}^{1} \text{ is inconsistent.}$$

At this point, the mechanical designer should evaluate another alternative and this time s/he proposes the plastic, which the manufacturing engineer *agrees* to use as the material for the headlight casing. The mechanical designer has finished with this branch of the design process but s/he still needs a report from the manufacturing engineer on the status of the manufacturing process.

9. Discussion

Several observations can be made from the above discussion. First, an essential requirement for effective design collaboration is to properly establish, maintain, and satisfy the dependency needs among design tasks of different designers in various stages of product development. For example, involving manufacturability objectives in designing front-end car structure reduces downstream design iterations and increases the design efficiency. In this paper, we took a decision-based approach to model the local design process and represented the dependency links as connections between those design decisions. This approach is different from other decision-based design models [4,5,12,13], since it provides understanding of the potential dependencies while designers are framing their decision problems. Moreover, it provides a global decision network that captures the effects of every decision made by a single designer on the total design process.

Second, the importance of identifying two distinct levels of coordination was illustrated. In value level coordination, designers coordinate the collective approach to the design problem by considering each others' values and preferences. In result level coordination, designers coordinate the details of a specific design alternative to select a common solution from a set of possible solutions for a connected problem. Once value level coordination is properly executed, the need to result level coordination will be minimized, since each designer has a relatively clear picture of dependent designers' values and strives toward generating design alternatives, to maximize the overall gain of the design team.

Third, agents play a critical role in facilitating

concurrency and maintaining dependency links among team members. As shown in above example, mechanical designer initiates coordination with the manufacturing engineer by proposing one alternative for designing the casing. Once the proposal was received by the manufacturing designer, two agents worked together to overcome the dependencies and propagate status of changes or updates among team members.

10. Conclusions and Future Work

In this paper, a decision network framework (ADN) for concurrent engineering design was introduced. The notion of design values was presented and the significance of early coordination of design values to achieve efficient collaboration was illustrated. Two major aspects of collaborative decision-making process [i.e. local decision-making (DDPM) and objective-based negotiation (OBNM)] based on design values were discussed. ADN framework was modeled and simulated using an agent-based design environment and a case study was created to examine the validity of the proposed framework. The results indicated that ADN framework increases the efficiency and effectiveness of the design process and reduces downstream design iterations comparing with traditional collaborative design processes. Further investigation is required to study the impacts of using different alternative generation methods and selecting the best alternatives from the alternative set. The tradeoff between selecting another alternative if a conflict arises and negotiation until the conflict is resolved needs to be investigated.

Acknowledgment

This research project was primarily supported by two National Science Foundation awards under grants DMI-9734006 and DMI-9726836. The authors are grateful to NSF for its support. Authors also would like to thank the reviewers for their insightful reviews and comments.

References

1. Barbuceanu, M. and Fox, M. S. 1994. "Conflict management with an authority/deniability Model," Proceedings of AAAI-94 Workshop on Models of Conflict Management, AAAI Technical Report WS-94-04, Seattle, WA.
2. Decker, K. S. and Lesser, V. R. 1995. "Designing a Family of Coordination Algorithms," UMass Computer Science Technical Report 94-14, University of Massachusetts, Amherst, MA.
3. Danesh, M., R. and Jin, Y. 2000. "An Aggregated Value Model for Collaborative Engineering Decisions," Proceedings of the Fifth ASME Design for Manufacturing Conference, Baltimore, MD.
4. Hazelrigg, G. A. 1998. "A framework for decision-based engineering design," Journal of Mechanical Design, Vol. 120, pp. 653–658.
5. Herling, D., Ullman, D. G., and D'Ambrosio, B., 1995. "Engineering Decision Support System (EDSS)." Proceedings of the Design Engineering Technical Conferences, September 1995, Boston, MA.
6. Howley, B., Cutkosky, M., and Biswas, G., 1999. "Compromising and Sharing Dynamic Models in an Agent-Based Concurrent Engineering Environment." Proceedings of the American Control Conference, Vol. 5, pp. 3147–3153.
7. Jin, Y., and Levitt, R. E. 1993. "I-Agents: Modeling organizations problem solving in Multi-agent teams," Intelligent Systems in Accounting, Finance and Management, Vol. 2, pp. 247–270.
8. Keeney, R. L. and Raiffa, H. 1976. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs.* New York: John Wiley & Sons, Inc.
9. Liu, J. and Sycara, K. 1994. "Distributed problem solving through coordination in a society of agents," Proceedings of the 13th International Workshop on Distributed Artificial Intelligence, Seattle, WA.
10. Malone, T. W. and Crowston, K. 1991. "Towards an Interdisciplinary Theory of Coordination," MIT Sloan School working paper #3294-91-MSA.
11. Medina-Mora, R., Winograd, T., Flores, R. and Flores, F. 1992. "The action workflow approach to workflow management technology," The Information Society, Volume 9, Number 4, pp. 391–398.
12. Mistree, F. and Allen, J. K. 1997. "Optimization in decision-based design," Position paper: Open Workshop on Decision-based Design, Orlando, Florida.
13. Mistree, F., Smith, W. F. and Bras, B. 1993. "A decision-based approach to concurrent engineering," Chapman & Hall, New York, pp. 127–158.
14. Pahl, G. and Beitz, W. 1996. *Engineering design – A systematic approach,* second edition, Springer, London.
15. Prasad, B. 1996. *Concurrent Engineering Fundamentals,* Vol. I and II. Prentice Hall, USA.
16. Petrie, C. 1993. "The *Redux'* server," Proceedings of the international conference on intelligent and cooperative information systems (ICICIS), Rotterdam.
17. Simon, H. A. 1969. *The sciences of the artificial,* MIT press, Cambridge, Massachusetts.
18. Smith, R. G. 1980. "The Contract Net Protocol: High-Level communication and Control in a Distributed problem Solver," Readings in Distributed Artificial Intelligence, pp. 357–366
19. Suh, N. P. 1990. *The principles of design,* Oxford University Press, Oxford.
20. Sun, J., Zhang, Y. F. and Nee, C. Y., 2000. "Agent-Based Product Design and Planning for Distributed Concurrent Engineering." Proceedings of the 2000 IEEE International Conference on Robotics and Automation. San Francisco, CA
21. Tan, G. W., Hayes, C. C. and Shaw, M., 1996. "An Intelligent-Agent Framework for Concurrent Product Design and Planning," IEEE transactions on Engineering Management, Vol. 43, No. 3., pp. 297–306.
22. Von Neumann, J. and Morgenstern, O. 1953. *Theory of games and economic behavior,* 3rd edition, Princeton University press, Princeton, New Jersey.
23. Yongtong, H., Ping, L., Yuhong, Y., Danian, Z., Changchao, M., Bode, J. and Shouju, R., 1996, "A Multi-Agent System for the Support of Concurrent Engineering." IEEE International Conference on Systems, Man and Cybernetics, Vol. 2, pp. 959–964.

Mohammad Reza Danesh

Mohammad Reza Danesh is a Ph.D. student of Mechanical Engineering at the University of Southern California. He received his Masters degree in Manufacturing Engineering from University of Southern California and Bachelor's degree in Mechanical Engineering from Sharif University of Technology in Iran. His research interests include mechanical design methodologies, manufacturing processes, concurrent engineering, collaborative decision-making, and agent-based support tools for concurrent product development.

Yan Jin

Dr. Yan Jin is Associate Professor of Mechanical Engineering at University of Southern California and the Associate Director of the USC IMPACT Laboratory. He received his Ph.D. Degree in Naval Engineering from the University of Tokyo in 1988. Since then, Dr. Jin has done research on collaborative design, multi-agent systems, and organization modeling. Prior to joining USC Faculty in the fall of 1996, Dr. Jin worked as a senior research scientist at Stanford University for 5 years. His current research interests include design process modeling and agent-based collaborative engineering. Dr. Jin is a recipient of 1997 NSF CAREER Award. He is currently heading a research program at IMPACT Lab to build knowledge infrastructure for collaborative engineering.