Intelligent Information Infrastructure for Group Decision and Negotiation Support of Concurrent Engineering

Katia Sycara and Michel Roboam[1, 2]

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Abstract

Concurrent Engineering is the activity that results in the design of a product taking into consideration downstream concerns, such as manufacturability, testability and maintainability of the designed product. Concurrent engineering is an example of group decision making and negotiation that commonly occurs in organizations. It involves many agents (e.g, design, manufacturing, materials, marketing), each of which has its own feasibility and performance criteria. Conflicting goals and viewpoints that arise during the process are resolved through *negotiation*. In this paper we introduce the concept of Intelligent Information Infrastructure to support the integration of distributed decision making and negotiation among the heterogeneous functions of a manufacturing enterprise.

1. Introduction

Traditionally, the production of a product by a manufacturing organization consists of a serial set of steps where the input of one manufacturing function is the output of the preceding one. For example, the design department designs the product and hands it to manufacturing. Manufacturing considers the design and discovers that various parts are difficult or even impossible to manufacture as designed. So, manufacturing engineers red-pencil the design drawings and send them back to the design department to be fixed. This practice, usually results in numerous Engineering Change Orders (ECO's) that lengthen the product life cycle, thus making it difficult for the enterprise to respond quickly to changing demand. The process of generating and implementing ECO's is extremely cumbersome, since it involves not only strictly technical issues, such as what changes should be made in the design, but also a host of coordination issues, such as design ownership, authorization of changes, acceptability of changes by the concerned parties, communication of the changes (especially problematic if the various concerned departments are geographically distributed), monitoring of changes and keeping track of design versions.

It has become increasingly obvious that for manufacturing firms to be successful in a rapidly changing and globally competitive environment, they must be able to:

- react rapidly to changes in fundamental requirements of products and to new technological opportunities
- minimize transition from design concept to product
- produce high quality products at the lowest possible life-cycle (design, manufacturing, testing, operation, maintenance) cost

The design methodology where downstream concerns, such as manufacturability, maintainability and testability of the product are taken into consideration in the early design stages is called *concurrent engineering*. Concurrent engineering reduces the need for ECO's, thus considerably shortening the product life cycle. Because concurrent engineering involves concurrency and interleaving of decisions, it requires tighter coordination that the serial decision making process of traditional design and production. Unless this tighter coordination can be effectively supported through the use of computerized coordination aids, concurrent engineering runs the risk of being even more inefficient and error prone than the serial process.

The process through which decision making is coordinated in concurrent engineering is a generalized negotiation process. We call it generalized negotiation because it exhibits both

[1]Michel ROBOAM is currently visiting scientist in the Center for Integrated Manufacturing Decision Systems and is sponsored by the AEROSPATIALE Company (France).

[2]This research has been supported, in part, by the Defense Advance Research Projects Agency under contract #F30602-88-C-0001.

integrative and distributive aspects. Traditionally, negotiation has been considered in a distributive context where the goals, values and interests of the parties are in conflict [13]. However, there are many situations, where integrative types of negotiation occur. In integrative negotiation, the parties engage in a group problem solving process characterized by increased cooperativeness and consensus seeking through increased information sharing, adaptation and problem restructuring [10]. Concurrent engineering is an example of such a process. It exemplifies situations which, albeit not strictly competitive, admit subgoal conflicts that could benefit from negotiation. In other words, although the overall situation is cooperative, conflicts might arise on ways to achieve a common goal, or on evaluation criteria. Such situations are very common in organizations where, although the various departments have common high level goals in terms of the particular tasks they are engaged in, conflicts also frequently arise.

In this paper, we report on the design and implementation of a comprehensive group decision and negotiation support system that supports group decision making in situations that admit varying degrees of cooperativeness. In particular, we focus on the intelligent information infrastructure part of the system, called EMMA (Enterprise Management and Modeling Associate). The intelligent information infrastructure constitutes a key aspect of a generalized negotiation support system in that it: (1) provides knowledge-based modeling support for the organizational entities and organizational context in which group problem solving and negotiation occur, (2) it allows for *distributed problem solving* in the sense that the parties engaged in group decision and negotiation can be geographically dispersed, (3) it supports multiple group decision making applications, (4) it enables the creation and maintenance of *distributed* model-, data- and knowledge bases that can be used in a group problem solving session and carry information between sessions, (5) it makes use of heuristic rules to automate access to information that may reside in distributed knowledge and data bases and disseminate the information to appropriate users without tedious manual intervention, (6) it supports task assignment and task monitoring, since it automatically can keep track of tasks and deadlines and alert the user, and (7) it enables the seamless integration of humans and expert systems as members of the group.

2. Overview of Concurrent Design

The design problem has the following characteristics:

- The global goal is to produce a design that is synthesized from contributions of different expertise, concerns and constraints
- During the design process, conflicts in the form of constraint violations could arise. If these conflicts are not resolved in a satisfactory manner, infeasible designs will occur.
- Disparate evaluations of (partial or complete) designs could surface as a result of different criteria used to evaluate designs from different perspectives. Typically, these criteria cannot be simultaneously and optimally satisfied. The design decisions that optimize one set of criteria could conflict with those that optimize another set. If these conflicts do not get resolved in a satisfactory fashion, design suboptimalities occur.
- The system goal is achieved by making the best tradeoffs on conflicting design goals and constraints.
- Because of the presence of conflicting constraints, goals and possibly evaluation criteria, it is impossible for each agent/expert/perspective to optimize the overall design using only local information.

As a result of the above characteristics, the final successful design can be viewed as a *compromise* that incorporates tradeoffs such as cost, ease of manufacturing and assembly, reliability and maintainability. We suspect that such compromises are commonly done implicitly by human design experts tacitly using rules of thumb (e.g., imprecise versions of other agents'/perspectives' evaluation functions). Typically, these implicit compromises go unrecorded making it very difficult to trace and avoid suboptimalities in the design. Because the proposed system incorporates knowledge-, model- and data bases and supports distributed information dissemination and update, it facilitates explicit recording of design proposals, modifications of design decisions, and associated justifications or objections. These constitute valuable information that (1) can help in system diagnosis and maintenance, and (2) can be re-used for redesign and reverse engineering.

The practice of concurrent engineering necessitates

increased coordination among the various functions of an enterprise. Decisions have to be made cooperatively, and conflicting constraints have to be resolved through negotiation. In design, negotiation occurs recursively at all levels and stages of design from conceptual design through embodiment to detailing and manufacturing of the artifact. The design is done by a number of design teams each of which contains various specialists and is responsible for producing a part of the desired artifact. For example, a detailed study of aircraft design [2] has found that aircraft design proceeds by the cooperation of specialists, each of which have their own model of the design. Design decisions are negotiated by the specialists among themselves. In aircraft design there are many specialists each with their own technology and language. For example, there are aerodynamicists who use surface models and equations, there are maintainability engineers, concerned with access, disassembly and replacement, there are hydraulic engineers, stress engineers, and thermodynamic experts. Failure to reach agreement on the design of a part that is the responsibility of one team is a problem not only for the particular team but for the manufacturing organization as a whole. Such failure must be communicated to other teams and resolved by negotiation.

Negotiation enters the design process at the following points:

- When different relevant specialists have made conflicting recommendations regarding values of attributes of a design.
- When an attribute value proposed by one specialist makes it infeasible for another specialist to offer a consistent set of values for other attributes.
- When a design decision made by one expert adversely affects the decision optimality of other experts.
- When alternate approaches can achieve similar functional results.

To support the process, we are currently developing a General Group Decision and Negotiation Support System for organizational decision making that can intelligently route information, allow access and update of various kinds of qualitative and quantitative decision models, allows queries and feedback, and incorporate protocols that embody various cooperation and negotiation schemes. EMMA is a Distributed Artifiical Intelligence (DAI) system. The agents in the system can be both human or knowledge based systems that are autonomous problem solvers and participate in group problem solving and negotiation. A DAI system demonstrates group intelligence by incorporating effective knowledge and coordination strategies for group decision making. The system serves as an invisible manager that supports group decision making by human and machine agents and monitoring group performance. The system currently has six different rule-based machine agents: a machining agent, a process planning agent, an inventory agent, and three schedulers each of which is responsible for scheduling a particular work area. The design agent has not been automated yet. Its role is played by a human. Each machine agent resides in a separate workstation and communicates with the other automated agents through a computer network. EMMA consists of additional software modules that have been implemented on top of the conventional network software to allow exchange and *machine interpretation* of messages, user transparency of information searches, access and update of distributed model bases, support of a variety of coordination strategies and support for organizational modeling.

3. Next Generation of Distributed GDSS/NSS

In this section we first present a brief assessment of the state of the art of GDSS/NSS systems and describe a set of additional capabilities that have been recognized by the GDSS/NSS community as needed in future systems. We, then give a brief description of the additional capabilities embodied in EMMA. Section 4 describes in some detail EMMA's architecture.

Many early GDSS systems were task driven, as defined by [5]. They were designed to meet the needs of one group performing one task, and therefore addressed one and only one application of group decision making. For example, one early GDSS was designed specifically to assist in labor management negotiation and could not be used for any other task [7]. More recently the need to provide a toolkit similar to the concept of tool set [11] has become apparent. Toolkits are collections of specific tools that address various tasks during group problem solving, such as information exchange, model management and model sharing.

Most GDSSs presented to date serve as a communication blackboard on which ideas can be generated, information can be shared and consensus may be reached by using group techniques such as voting and preference ranking [8]. This type of system can provide valuable support to group decision making. However, in some situations, a higher level of support may be needed especially when there are conflicts to be resolved [4, 8, 6]. For

example, when accounting, production amd marketing managers fail to reach agreement on the forecasted sales for next year, it is usually not a matter of voting or preference ranking. Nor will a multi-attribute decision model or game theoretic approach by themselves be appropriate for resolving conflict. In fact the disagreement may result from differences in assumptions, conflicting evaluation criteria, or in the selection of models. The disagreement over sales figures might have occurred in the context of the managers' working together to develop a strategic business plan based on the anticipated demand and available resources. The marketing manager may use product pricing, sales forecasting and market segmentation models. From the accounting perspective, however, pricing decisions may need to also consider costs. To production managers, the cost allocation and capital budgeting information must be integrated into capacity planning, production scheduling and inventory. It is obvious that in this process, different parties, representing different interests and corporate subcultures, having different factual information and value judgment, and using different decision models, need to communicate and negotiate to reach agreement.

A second example concerns engineering design. Consider the process of designing a turbine blade. Some of the dominant perspectives are aerodynamics, structural engineering, manufacturing and marketing. The concern of aerodynamics is aerodynamic efficiency; for structural engineering it is reliability and safety; for manufacturing, it is ease and cost of manufacturing and testing; for marketing it is overall cost and customer satisfaction. The two variables of concern in a turbine blade that we consider are: (a) root radius, and (b) blade length. From the perspective of structural design, the bigger the root radius, the better since it decreases stress concentration. From the perspective of aerodynamics, the smaller the root radius, the better, since it increases aerodynamic efficiency. Concerning the length of the blade, from the point of view of structural design, the shorter the blade, the lower the tensile stresses; from the point of view of aerodynamics the longer the blade, the better the aerodynamics. On the other hand, if the blade is shorter, it makes for a lighter engine which is a desirable characteristic for aerodynamic efficiency. From the point of view of marketing, aerodynamic efficiency lowers the cost of operation of the aircraft, thus making it more attractive to customers. From the point of view of manufacturing, it is easier to manufacture shorter blades with bigger root radii.

To reconcile these conflicting concerns, typically human specialists representing the various perspectives have series of meetings where they discuss and negotiate the assignment of appropriate values to sets of design parameters. Each team could include individuals from different parts of the organization. The organizational areas influence the team members perspective on the discussed concepts, and relative importance of concepts and issues. Hence multiple and conflicting interpretations of the task description could surface.

During the decision processes illustrated by the two examples, the agents not only use quantitative and qualitative models to facilitate their analyses but also must convince other group members that the results generated by their models are of value. The accounting and production managers in the first example, may question the forecasted sales estimated by the marketing department. The marketing manager may want to know how the production process is scheduled and what criteria are used in the capital budgeting process. Thus a higher level of support is needed.

The need for intelligent management of quantitative models in group decision support has been identified in the literature (e.g., [8]). The consideration of quantitative models, however, must be integrated with knowledge based models [1] to facilitate the explicit examination and manipulation of relations that are implicit in the quantitative formalisms, to allow the communication of the agents' goals, assumptions, evaluation criteria, justifications and organizational context. Thus, support is needed both in facilitating the development of models (both quantitative and knowledge-based) and in enabling access, execution, and sharing of such information. In the production example, it would be useful for the system to provide functions that not only allow the managers to examine what models were used to generate their sales figures, what assumptions were behind these models, and how these models were evaluated, but also allow the managers to communicate to each other their conclusions, questions and justifications concerning the models and other aspects of the task.

In group problem solving, there are two generic bases for disagreement: uncertainty due to incomplete information and conflict objectives. So, a General Group Decision and Negotiation Support System must facilitate communication in order to simplify model utilization and information sharing. The information shared could be numeric, textual, and relational and is used to reduce disagreement caused by incomplete information. Models (both quantitative and qualitative) can be executed and the input and output information can be disseminated to interested members.

Most existing GDSSs support groups that are

located in the same room in face-to-face meetings. Such systems support the display of suggestions by the group on screens visible by the group members. Thus the screens serve as means of simultaneous access to shared information. Very often, however, members engaged in group problem solving are geographically distributed. For example, in large companies, the design, manufacturing and marketing departments may be in different cities or states. Different time zones may make it difficult to hold teleconferences, thus necessitating distributed asynchronous processing. In other words, the users do not need to be present in the same location at the same time. In addition, the users can process the information sent to them or make information request in an asynchronous manner, i.e. at their convenience without having to interrupt ongoing work.

The distributed environment that we have implemented has a variety of general characteristics. First, information dissemination is done via communication of messages that are *machine interpretable*, thus facilitating information monitoring and user notification. Second, multiple users have access and share multiple models. In our distributed environment, each user has access to to an individual DSS. Each individual DSS contains models and data owned by a particular user. There is also a global/group knowledge base that stores information accessible to all individuals. The global knowledge base includes the organizational model that defines the roles, responsibilities, and interrelations of the agents in the organization as well as coordination strategies and models designed to support group activities such as nominal group techniques and brainstorming processes. Third, during system use, the location of information is transparent to the user. In other words, the distributed system allows members to access models stored in an individual model base without the need to go through a tedious manual search process. Fourth, agents in the system can be human or AI systems. Users can query the local or global knowledge and data bases. The query processing function serves as an interface through which the users place their requests. The module labeled "system" translates a query into machine-translatable commands. Our system uses a structured query language which is an adaptation of SQL. The addition of knowledge bases enables decision systems to provide support for integrating information across sessions and between groups. System output can be presented both in text and graphical form through the utilization of domain-independent graphical rendering tools that have been developed in our laboratory. The following figure presents the overall system architecture:

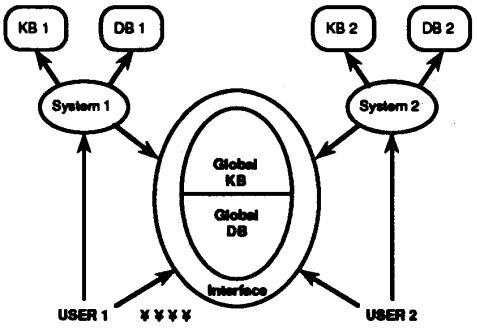

Figure 3-1: EMMA Design

4. The Design of EMMA

Concurrent engineering is a group decision making process where varying degrees of conflict and cooperative behavior are exhibited on the part of various organizational entities. Central to the group decision making and negotiation for conflict resolution is the processing of information [3]. Information processing includes the sequencing of requests for information, the specification of assumptions within each request, and the manipulation of available information. Automating information processing for each of the organizational functions as well as automating the coordination process affords integrated decision making support.

EMMA is a system under development to provide such support. It is a hybrid human-computer decision making system in which human users instigate the decision making process and in which the system engages in distributed problem solving to support the users. Since the distributed computer system manages a network of experts, it may be viewed as a sort of collective expert with which human users interact in the course of group decision making and negotiation. There are several information communication capabilities that EMMA provides:

- *Information routing*: given a symbolic representation of a message to be distributed and a representation of the goals and information needs of group members, the information routing capability provides for (1) transferring information to groups where the sender and the receivers are pre-defined, (2) transferring information to groups which appear to be interested in the information. This is accomplished by matching a group's goals and information needs to the information packet. Thus information is automatic without the explicit manual intervention of the user. (3) reviewing old

information to see if it matches new goals and information requirements specified by a group.

- *Closed loop system*: Initiation of activities resulting from information dissemination may necessitate providing feedback to the initiator. Heuristics trigger the feedback process.
- *Command and control*: Given a knowledge based model of the organization which includes personnel, departments, resources, goals, constraints, authority and responsibility relations, the system supports these lines of authority and responsibility. For example, if negotiation for a task is proven fruitless, the agents can access the organizational model to find an arbitrator with the appropriate authority.

The design of the system is layered with each layer of software relying on the functionality provided by the previous layer. The six layers that comprise the architecture are: the Network layer, the Data layer, the Information layer, the Organization layer, the Coordination layer and the Market layer.

The **Network Layer** provides for the definition of the network architecture. At this level, the automated agents are named and declared to be part of the network. Message sending (or message passing) between agents is supported along with synchronization primitives. The functionality of this layer is a sophisticated extension of computer network functionality in that it allows the communication of structured messages of various kinds that are machine interpretable.

The **Data Layer** provides for queries and information management of distributed knowledge-bases and models that are associated with each agent. The query language used in the system is a formal query language patterned after SQL.

The **Information Layer** provides "invisible" access to information spread throughout the system. The goal is to make information located anywhere in the system locally accessible without requiring the agents to know where the information is located.

The **Organization Layer** provides the primitives and elements (such as goal, role, responsibility and authority) for distributed problem solving. It allows automatic communication of information based upon the roles an automated agent plays in the organization. Each agent knows its place, its responsibility, its goal, and its role in the enterprise organization.

The **Coordination Layer** provides protocols for group decision and negotiation support for the agents in the decentralized system.

The **Market Layer** provides protocols for coordination among organizations, such as suppliers, customers, subcontractors, in a market environment. It supports the distribution of tasks, the negotiation of change and the strategies to deal with the environment.

4.1. Single Agent Architecture

Each agent in the system is either a human or an expert system that consists of the following subsystems: (1) Problem Solving Subsystem, (2) Knowledge Base, (3) Knowledge Base Manager, and (4) Communication Manager.

The **Problem Solving Subsystem** consists of the inference engine and decision models which allow the agent to solve problems related to its domain of expertise. The local execution cycle is triggered either by the internal transactions generated during local problem solving, or by external events forwarded to the agent by the Communication Manager.

Each agent contains a locally maintained object-oriented **Knowledge Base** to support its problem solving. The Knowledge Base maintains descriptions of various entities, such as physical objects (products, resources, operations, etc) or conceptual objects (customer orders, process plans, communication paths, temporal relations, etc). The design of the knowledge base relies upon artificial intelligence representation concepts and techniques for the manipulation and storage of models in the system. In particular, we make use of semantic inheritance networks and frame representation.

The **Knowledge Base Management System** manages information exchanges between the problem solving subsystem and the knowledge base, maintains the consistency of the local knowledge base, and responds to requests made by other agents. In the distributed decision support system, knowledge and data may reside anywhere in the system and can be transparently accessed. Knowledge falls into one of two classes: that *owned* by an agent which must be stored locally, and knowledge *used* by an agent which can be stored at another agent and a copy may be stored locally.

During decision making, it might not be possible to find needed knowledge in an agent's local

knowledge- or data base. The agent may or may not know where in the system the knowledge may reside. It is the responsibility of the Knowledge Base Management System to "hunt down" the missing knowledge and to respond to requests from other agents. To accomplish this, the Knowledge Base Management System includes a **Communication Manager**. The Communication Manager manages the search for information in the distributed system and responds to requests from other agents. To perform these activities, the Communication Manager has two modules:

- The **searcher** corresponds via message sending with other agents. The searcher performs two tasks: searching for knowledge not available locally, and the updating of knowledge changed and owned by the IN-node.
- The **responder** answers messages originating from other agents' searchers, and updates the local knowledge base according to updating messages.

4.2. The Network Layer

The Network Layer defines the network structure. The nodes of the network are intelligent machine agents. In the example of concurrent engineering of a turbine blade presented in the introduction, the problem solving agents would be marketing, manufacturing, aerodynamics, and structural engineering. In order for the nodes to cooperate and negotiate, they (a) must be aware of the existence of the other agents, and (b) must be able to interconnect and communicate with them. These main functions are performed by the Network Layer. The Network Layer contains a variety of elements:

- *Nodes* represent problem solving agents (expert systems), such as different design specialists, a manufacturing specialist, a materials specialist, etc.
- *Channels* define communication links between agents.
- *Messages* can be sent along channels between agents. Message passing functions must support these exchanges.
- *Protection* is provided so that messages can only be processed by legal agents.
- *Synchronization primitives* are defined to synchronize the internal problem solving of an agent and communication activities.
- *Communication primitives* such as "message-passing" and "message-reception" are specified to support message exchanges between agents.

4.3. Data Layer

The Data Layer provides agents with the capability to explicitly request and send information from/to other agents. The protocol for requesting and asserting information between agents is based on a subset of SQL. Protection and locking mechanisms are provided to ensure consistency. Since agents may have different representations for different pieces of information, the system includes a dictionary and a common communication language to promote mutual understanding.

4.4. Information Layer

The information layer supports the information exchange between the various agents. We have identified and implemented two functionalities at this layer: automatic information acquisition and automatic information management.

The acquisition of information starts from a specific need of a specific agent. Each agent must have the capability to acquire at any time all the knowledge it needs to solve a particular problem. If a particular piece of information is not localy available, the agent generates an information request to agents who may have the knowledge. The utilization of each piece of information by its users is done according to some specific views and privileges which are defined by the information *owner*. The knowledge that is created by an agent belongs to this agent. The owner of information is the only one allowed to add, remove or change it. In addition, the owner of a piece of information must define for this piece of information the agents that are allowed to access this information and the type of accessing rights that each will have. The accessing rights that are implemented in the system are: select, insert, delete and update information. These rights are associated with every piece of information.

Information acquisition occurs automatically, when an agent's problem solver attempts to access information that does not exist in its knowledge base. Four methods are then used by the knowledge base manager to acquire the information. First, the owner of the needed information may be another agent. Therefore it is reasonable to believe that the owner agent may have the desired information. Secondly, the semantic inheritance network contains pointers to those agents that maintain information of a particular type. Third, the agent maintains a list of agents that it corresponds with on a regular basis and may query them. Fourth, as a last resort the request may be

broadcast to all agents.

Automatic information management maintains the consistency of an agent's knowledge throughout the distributed system. Consistency maintenance is done through information updating and information dissemination. Information updating maintains consistency in the knowledge bases of the agents. Information distribution sends new knowledge to agents that are potentially interested. The questions are what, when and to whom should information be distributed ? The system has lists of potentially interested agents for each piece of information generated in the system. For example, when the design agent makes a design decision that involves machining tolerances, this piece of information is automatically sent to the machining agent, since the information has been tagged as being of interest to the machining agent.

4.5. Organization Layer

At this Layer, the definition of roles, responsibilities, authority and goals specific to each agent is added to the system. Hence, each agent knows its role, function and authority relationships in the organizational structure. The organizational model of a manufacturing enterprise that we build at this layer concerns the definition of the interactions and relationships which exist between the members of a manufacturing system. Such a system can be split up into three subsystems:

- The **physical** subsystem which includes the people, the machines, the material flows, etc of a manufacturing system.
- The **decisional** subsystem which controls the physical system by triggering and readjusting its activities. We introduce the concept of operating level which links the decisional and physical level (it includes the control of machines, the security procedures, etc.).
- The **informational** subsystem which corresponds to all the information and information processing that can occur between or inside of the two previous systems.

The organizational model must show both aspects of goals and hierarchical interdependencies. It must also describe decision making processes throughout the organization. All these constraints have oriented our choice toward the GRAI methodology [9].

The domain covered by the GRAI method concerns decisional systems on the two first levels of abstraction. This domain is represented in the GRAI

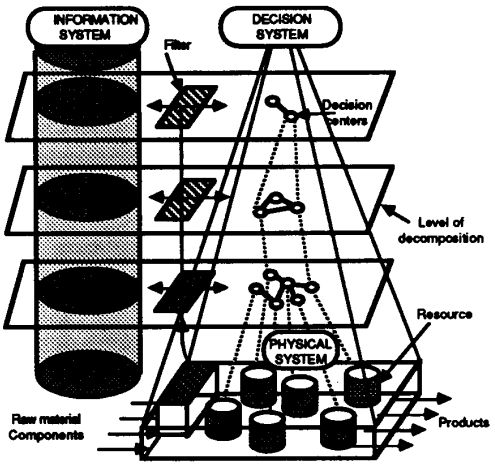

Figure 4-1: Global conceptual model of GRAI

method by conceptual models which allow the identification of its main characteristics. Two models are specified:

- The global conceptual model: it specifies the elements of a manufacturing system and defines the place of the decisional subsystem in this organization. The decisional subsystem is represented as a hierarchical set of decision centers (figure 4-1).
- The detailed conceptual model defines the content of a decision center (figure 4-2). In this model the concept of decision frame is defined. It represents the goal transmission and hierarchical links between two decision centers. In addition the concept of decision variables is specified. It represents the elements on which a decision center can act to make its decisions (for example, the activity "to adjust the load planning" can be done by acting on the "time", "the resources", "the capacity", etc (all these elements are decision variables and are given by another decision center).

The GRAI method has two different graphical tools used for modeling decision subsystem of manufacturing systems. The first tool is called the **GRAI grid**. The second tool are the **GRAI nets**.

The GRAI grid gives a global and hierarchical representation of the whole decision system of a manufacturing system. More precisely, it describes the production management system of a manufacturing organization by defining the decision

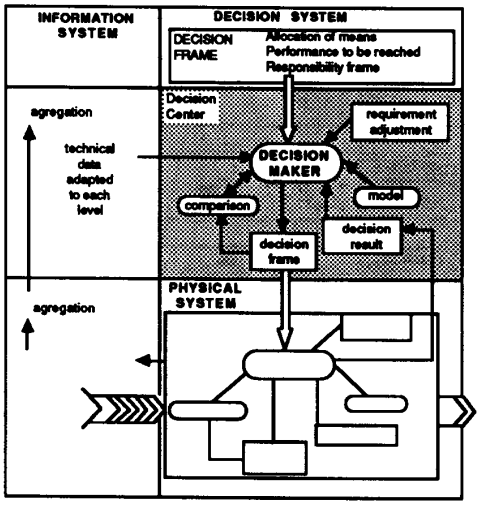

Figure 4-2: Structure of a Decision Center

activities performed in this system. The grid has two axes: (1) The horizontal axis indicates the **functions** of a production management system. Classically, we identify the planning function, the purchasing function, the quality control function, the engineering function, etc. (2) The vertical axis defines a **temporal decomposition** of these functions.

Each "box" defined in this grid represents a decision center at different levels of abstraction (for example, "to make the Master schedule", "to make the schedule", "to define the supplying parameters", etc.). In addition, we add on this grid links between the decision centers. The GRAI nets define the content of a decision center. Each "box" of the grid is decomposed into a GRAI net (or several, according to the level of detail the user intends to reach) which describes the sequence of activities performed in it. It also defines the elements, information, resources, etc used to perform each activity.

4.6. Coordination Layer

The Coordination Layer provides the protocols for the problem solving activities between the agents of the decentralized system. We present an example scenario of concurrent engineering to illustrate the functionality of the coordination layer and the layers beneath it [12]. The agents interact through message passing. The messages that the negotiating agents exchange contain the following information:

- The proposed design
- Justifications of design decisions
- Evaluation of a proposal from an agent's point of view
- Agreement or disagreement with the design proposal
- Requests for additional information, such as with which issue in the proposed design the agent disagrees
- Reasons for disagreement
- Arguments in support or against a proposal

The following is a simplified example dialogue of the various concerned agents in an attempt to arrive at a mutually satisfactory turbine blade design:

Aerodynamics (possibly reasoning from existing designs) suggests to structural engineering, and manufacturing particular values x and y for length and root radius of the blade. The suggested x and y values are within acceptable constraint ranges.
[*This requires that Aerodynamics make use of the Organizational Layer to find out which of the organizational functions it needs to communicate this information to. After this information has been obtained, the Network Layer is used to find out the names and addresses of these agents so information can be sent to them. Existing designs might not reside in the local knowledge base of the Aerodynamics agent. The Information layer transparently finds the requested information concerning length and root radius of the blade from where it resides.*]

Structural engineering evaluates these values from its point of view and suggests values x' and y' where x'<x and y'>y (i.e., shortening the length and increasing the root radius) to increase safety.
[*The Data Layer enables Structural to access its Problem Solving Subsystem to evaluate the proposal of Aerodynamics. The Coordination Layer is used to enforce the negotiation protocol that specifies that upon receipt of a proposal, an agent must evaluate it and, if it disagrees, suggest a new proposal*].

Aerodynamics rejects the values structural engineering suggested aying they would considerably decrease aerodynamic efficiency.
[*The negotiation protocol at the Coordination Layer specifies that if a proposal is rejected, reasons for rejection should be presented*].

Structural engineering counters that shorter blade makes engine lighter, thus also increasing efficiency.
[*the negotiation protocol at the Coordination Layer specifies that arguments in support or against a proposal should be presented*].

Marketing now enters the dispute saying aerodynamic efficiency sells the product since it is less costly to operate.
[*Marketing, which had not received direct communication from Aerodynamics is also alerted because the automatic information management function of the Information Layer uses its heuristics and lists of potentially interested agents finds out that Marketing might be potentially interested in the negotiation and sends it the Aerodynamics and Structural Engineering proposals and counterproposals*].

Manufacturing supports structural by saying it is easier to manufacture short blades with big root radius.

If negotiation is protracted without resolution, the agents might vote to consult an agent with higher authority to arbitrate. The Organizational Layer supplies information about potentially suitable arbitrators.

4.7. Market Layer

The Market Layer will support coordination among various organizations. Our work in defining this layer is still in the initial stages. We intend to define the role various enterprises play in the market, such as suppliers, customers, subcontractors, and model their problem solving behavior. Moreover, we want to study changes in the problem solving behavior of agents as a result of variations in supply and demand. In particular, we are interested in modeling the effects of changes in supply/demand patterns on coordination style.

5. Concluding Remarks

Concurrent engineering design is an example of the group decision and negotiation that commonly takes place in organizations. In order to support this process, we are developing a DAI system that manages the modeling and information needs of the group members (human and machine agents). This Intelligent Information Infrastructure constitutes a key aspect of a generalized negotiation support system in that it: (1) supports multiple applications, (2) allows asynchronous communication and information processing of groups of geographically dispersed agents, (3) supports the creation and maintenance of distributed decision and organizational structuring models, and (4) flexibly integrates problem solving of machines and humans.

References

1. Applegate, L., Chen, T.T., Konsynski, B., Nunamaker, J. "Knowledge management in organizational planning". *Journal of Management of Information Systems 3* (1987), 20-27.

2. Bond, A., and Ricci, R. Cooperation in Aircraft Design. Proceedings of the MIT-JSME Workshop on Cooperative Product Development, Cambridge, Mass., 1989.

3. Bui, T., and Jarke, M. "Communication requirements for group decision support systems". *Journal of MIS 2*, 4 (1986), 8-20.

4. DeSanctis, G., and Gallupe R. B.. "A Foundation for the Study of Group Decision Support Systems". *Management Science 33* (1987), 589-609.

5. Huber, G. "Issues in the Design of Group Decision Support Systems". *MIS Quarterly 8*, 3 (1984), 195-204.

6. Jarke, M. "Knowledge sharing and negotiation support for multiperson decision support systems". *Decision Support systems 2* (1986), 93-102.

7. Kersten, G. E. "NEGO - Group Decision Support System". *Information and Management 8* (1985), 237-246.

8. Liang, T.P. "Model Management for Group Decision Support". *MIS Quarterly 12*, 4 (1988), 667-680.

9. Roboam, M. *Modeles de reference et integration des methodes d'analyse pour la conception des systemes de production.* Ph.D. Th., Laboratoire GRAI, Universite de Bordeaux, Bordeaux, France, 1988.

10. Shakun, M., F.. *Evolutionary Systems Design: Policy Making Under Complexity and Group Decision Support Systems.* Holden-Day, Oakland, CA., 1988.

11. Sprague, R.H. "A Framework for the Development of Decision Support systems". *MIS Quarterly 4*, 4 (1980), 1-26.

12. Sycara, K. Negotiation in Design. Proceedings of the MIT-JSME Workshop on Cooperative Product Development, Cambridge, Mass., 1989.

13. Sycara, K. "Negotiation Planning: An AI Approach". *European Journal of Operational Research 46* (1990), 216-234.