Object-Oriented Manufacturing Systems

The seven articles in this special issue were originally presented at the International Conference on Object-Oriented Manufacturing Systems May 3–6, 1992, in Calgary, Canada. The criterion for selection was the presentation of a knowledge-based approach, architecture, or technique that is of wider applicability than the manufacturing domain for which it was originally described and that would be of interest to the larger knowledge-engineering community. Each of the selected articles has been extended and updated for publication in the special issue and independently reviewed.

Because the theme of the original conference was object-oriented manufacturing systems, naturally all of the research systems described in these articles were implemented using the object-oriented paradigm. Except for this, they differ widely in their focus. The subjects range from active objects to agent systems and knowledge repositories on through to failure recovery mechanisms. The architectures span from strictly hierarchical to completely decentralized. Communication processes go from simple message sending to complex message structures to multiple specialized blackboards. Control structures cover the range from localized to centralized. Above all, these articles show the variety of innovative knowledge-based approaches now being used for real-life application. The following gives an overview of each of these interesting articles.

Knowledge repositories are the future generations of present-day data bases, storing knowledge in an integrated or potentially integratable form. Probably, a combination of both central and distributed repositories will be needed. For this environment, B. Gaines in "Experience with a Class Library for Organization Modeling and Problem Solving" puts forward the concept of a lightweight knowledge representation server surrounded by small functional modules to support a variety of applications. The server is sufficiently lightweight that many instantiations of it can exist on a computer workstation yet powerful enough to cope with knowledge structures involving thousands of concepts and tens of thousands of entities. The server has capabilities for definition, assertion, subscription, recognition, constraint propagation, rule propagation, and so on, and there are functional modules for text analysis, hypermedia, repertory grid elicitation, conceptual clustering, problem solving by frame/rule/case-based inference, and others. Such a knowledge storing, processing, and integrating system can have potential application in any information-based environment. Its visual interface allows for continuous user management of the uncertainties, errors, conflicts, and changing circumstances within this environment.

Decentralized systems within which there is coordinated activity to solve a problem are attracting much attention these days. While hierarchical control systems with centralized intelligence can be well suited for similarly structured real-life applications, there are many situations where decentralized or partially decentralized control is advantageous. In "Scheduling Strategies for a Decentralized Object-Oriented CIM Control System," D. J. Bagert and D. Carrol describe a fully decentralized scheduling system with a single-layered architecture based on a pressure-flow model. Mobile objects move through a configuration of processing cells according to a "pressure differential" strategy. Each mobile object is responsible for, as it were, finding its own way through the system based on a process program that it contains. This scheduling system was found to be naturally fault tolerant and able to adjust to changes in its operating environment.

Although the concept of active objects dates back to the first object-oriented language (SIMULA), it is commonly only passive objects that are used in object-oriented programming today. T. Mionoura, S. Choi, and R. Robinson in "Structural Active Object Systems for Manufacturing Control" describe a whole programming system based on active objects, where each such object implements its own portion of control to initiate its operations (i.e., localized control). Moreover, these objects can be

Integrated Computer-Aided Engineering, 1(2) 91–92 (1993)

CCC 1069-2509/93/020091-02

"plugged" together using interface variables that act rather like terminals of hardware components. This enables structural and hierarchical composition of software components into larger entities and allows for interchangeable components. Such larger prototype systems have been developed for a range of applications, including manufacturing.

In manufacturing, as in many other application areas, simulation is increasingly being relied on for planning and prediction. One potentially promising approach to simulation is agent oriented. Such an approach is presented by E. Mayrand, P. Lefrancois, and B. Montreuil in "Agent-Oriented Simulation Model of Manufacturing Activities" for a hierarchical production planning and scheduling system. The agents are frame based, communicate through a blackboard, and respond opportunistically to changes in a blackboard. For rule processing, an agent is linked to a forward-chaining module. This system was applied to the scheduling of rolling mills that produce aluminum foil, with appropriate classes being defined for WorkOrder, Operation, Loader, JobManager, CommunicationManager, and so on.

A blackboard is often used as an information sharing and coordination mechanism in larger knowledge-based systems. In "Using a Public 3-D Gantt Chart Communication Structure between Agents for Distributed Scheduling Architectures," M. H. Jobin, P. Lefrancois, and B. Montreuil describe an unusual 3-D blackboard, a Gantt chart, that shares specialized scheduling information and provides a user interface. The sharing is among intelligent agents that derive ultimately from a special class called "Organism." The application of this research is for distributed scheduling and the work was initiated by problems faced by a large rolling-mill facility.

To integrate different systems into a larger functional system is a difficult task when each individual system has its own formats for inputs and outputs. There is much effort in computer-integrated manufacturing (CAM) directed to this problem and to data interchange standards that can alleviate it. In "Service-Oriented Integration Environment for CAD/CAM Modelers and Application Programs," S. P. Magleby, K. B. Gunn, and C. D. Sorensen suggest a different solution, which could also be applicable to other domains. The basis for this solution is to wrap up each computer-aided design (CAD) modeler, analysis program, or application as an individual object within an overall object-oriented environment. Because objects communicate by messages, the objects can be run as separate processes provided there is an adequate communication mechanism. The structure of information sent within a message from one object to another does not have to conform to any comprehensive standard—it is only necessary that each such pair of objects understands this structure. This system was implemented within a UNIX environment and can be extended easily with additional applications being added as objects.

To provide for graceful (or at least adequate) recovery from failure is the dream of every software or hardware designer. In modern manufacturing, where there is a complex flow of components through a structure of expensive automated production cells, the ability to recover quickly from cell mechanical or electrical failures is economically important. Rescheduling around a cell may provide a temporary solution but rapid recovery of the cell is still desirable. In "Failure Recovery in Automated Manufacturing Cells through Opportunistic Reasoning," Y. B. Moon and C. L. Moodie describe an opportunistic failure recovery approach adopting a multiple-blackboard architecture. The four blackboards used are specialized respectively for information relating to diagnosis, remedies, supporting actions, and control. The underlying strategy is employing cause-based control to implement recovery. When a failure occurs, each possible-cause knowledge source searches for the facts that would support its case, and this information is then examined to determine the exact course. The corresponding cause knowledge source can then initiate (or recommend) an appropriate remedy.

Each of the articles included in this special issue is worthy of this wider audience in integrated computer-aided engineering. It is hoped that their publication will stimulate further innovation, research, and applicability in this fascinating field.

Douglas H. Norrie
Guest Editor
The University of Calgary
Calgary, Alberta, Canada

Note: Enquiries regarding the original proceedings volume for the International Conference on Object-Oriented Manufacturing Systems should be directed to the Division of Manufacturing Engineering, The University of Calgary, 2500 University Drive, Calgary, Alberta T2N 1N4, Canada.