A Multi-Agent Intelligent Design System Integrating Manufacturing And Shop-Floor Control

Sivaram Balasubramanian and Douglas H. Norrie
Division of Manufacturing Engineering
Department of Mechanical Engineering
University of Calgary
Calgary, AB, Canada T2N 1N4

Abstract

A multi-agent architecture has been developed for the integration of design, manufacturing, and shop floor control activities. This is based on cooperating intelligent entities in the sub-domains which make decisions through negotiation, using domain-specific knowledge both distributed among the entities and accessible to them. Using this architectural framework, an Agent Based Concurrent Design Environment system has been developed for feature-based design, manufacturability evaluation, and dynamic process planning. This is a multi-agent prototype system involving the following types of agent: design agent; geometric interface agent; feature agent; part agent; machine agent; tool agent; environment manager; and shop floor manager. A new technique for evaluating manufacturability is introduced, based on interacting intelligent features of the part being designed. This proof-of-concept system was developed for three-dimensional prismatic parts, with twenty-five different feature types, but can be extended to other geometries. The system has been completed and tested, and is being integrated into a larger multi-agent environment incorporating routing, scheduling, and overall production control.

Introduction

Globalization of the economy and the resulting competitive industrial environment demands that the industry develop new products with shorter lead times and better overall performance. Product design decisions play an important role in determining the form, cost, reliability and market acceptance of the final product and hence the emerging concurrent engineering (or simultaneous engineering) design methodology aims to address various life-cycle values of a product in the early stages of its design [Ishii 1990]. Instead of following the conventional, sequentially arranged product development process, concurrent engineering design incorporates considerations such as manufacturability [Mahajan et al. 1993, Regli et al. 1994], assembly [Jackeila et al. 1985, Jackson et al. 1993], serviceability [Makino et al. 1989], and recyclability [Markino et al. 1994] early into the design phase. These aspects of the product are considered conjointly with the design functions. This approach can reduce the number of redesigns, thus shortening product development lead times. In addition, better products with more balanced overall performance can be produced, because of early consideration of all relevant aspects of the product life cycle.

Achieving good manufacturability and minimum production cost is often considered to be the most critical factors when implementing concurrent engineering design. This brings the downstream concerns up front and helps avoid costly redesign iterations at the manufacturing stage. Using Design for Manufacturability tools, a designer can significantly improve designs through trade-offs between functionality and manufacturability. Other considerations, such as assembly, maintainability, and serviceability can often be incorporated into the design phase in a similar manner. For effective concurrent engineering, software tools are needed for systematically collecting knowledge and data that are related to the design, manufacturing, service and maintenance tasks and for synthesizing optimal design with balanced overall performance. The two most important capabilities for concurrent engineering design software relate to the generation of all feasible design and to the evaluation of manufacturability and production costs jointly with design functional performance for each alternative design.

Concurrent engineering aims at quick product turnover and better balanced product, by integrating various issues of product life cycle right at the design stage. Though the philosophy and the need have been well studied, the implementation of the concept necessitates radically new approaches to achieve concurrency in the complex multi-objective decisions, based on the large volume of interacting domain-specific information. One solution is to use cooperating intelligent entities to represent domain specific knowledge and make decisions through a negotiation process. Such an implementation is possible by using the concept of a heterogeneous multi-agent system. This paper describes a feature-based design system for prismatic components, implemented within the framework of multi-agent based concurrent engineering design. The multi-agent architecture for integrating design, manufacturing and shop-floor control is also described and the implementation issues associated with this integration are analyzed.

Related Work on Feature Based Design

Feature-based design is central to the Agent Based Concurrent Design Environment developed. The major features of several key existing feature-based design systems are therefore briefly reviewed.

The First-Cut project is a feature-based design system in which individual features and their corresponding process plans are combined to create a part and manufacturing plan concurrently [Cutkosky et al. 1988]. The Next-Cut feature-based design system was developed to work on designs from more than one level of detail and across multiple views while maintaining consistency of part representation [Cutkosky et al. 1992].

The Arizona State University Features Testbed is a system for designing, documenting and evaluating parts and is organized into a shell for product definition and another for mapping and applications [Shah and Rogers 1989]. The Feature Modeling Shell includes a form feature modeler as well as other modelers for tolerances, materials, etc. Each modeler can be used in setup mode to create specific features and feature knowledge. These feature libraries can then be used to create part models.

The Quick Turnaround Cell System (QTC) [Anderson and Chang 1990] combines a feature-based design system with automatic process planning. In this system, the features are related to machining operations, but do not correspond on a one to one basis with manufacturing features. CADETS is a feature based design system incorporating manufacturing constraints [Wright and Hannam 1989].

For products designed in an external location the sequential process of feature recognition, process planning and scheduling may be quite appropriate and one such approach is given in [Gu et al. 1994].

A general architecture for the design with features concept is given in [Dixon 1988]. A comprehensive review of early research in mechanical engineering design including features and feature based design can be found in Finger and Dixon [1989a, 1989b]. A more recent summary of research in feature based design is given by Salomons [1993].

The following conclusions can be drawn from the literature review. Feature identification and definition is necessary to adequately represent tolerancing and other manufacturing related information. There are two basic approaches: recognize (identify) appropriate features after designing with geometric elements or to design from the beginning using features. Despite the considerable improvement in feature recognition techniques, they do not produce a sufficiently complete set of data and certain manufacturing related information has to come from other sources, thus making complete automation difficult. Moreover, feature recognition is a sequential process and hence, concurrent evaluation of features for manufacturability is not feasible. Feature based design does not have these disadvantages, but can have the problem of having too many features to easily manage. User configurable features provide a feasible solution to this problem. However, feature-based design representations are non-unique, depending on the selection and use of features. Concurrent automation is possible with feature based design and hence, holds more promise for concurrent engineering design.

The initial approaches to concurrent engineering design used a centralized knowledge base. A distributed approach is advantageous since information pertaining to different domains can be separately and better organized and multi-objective decisions can be achieved through the cooperating entities accessing these knowledge bases. Such an approach is feasible through the concept of heterogeneous multi-agent systems.

In recent years, several distributed agent approaches to concurrent engineering have been reported, including the large-granularity Palo Alto Collaborative Testbed (PACT) [Cutkosy et al. 1993] in which existing design and other systems were 'wrapped' to form agents and the finer-granularity Design world system [Genesereth 1991] in which processes were implemented as agents.

A Multi-Agent Framework for Concurrent Engineering

Materials and shop floor resources greatly affect the manufacturability analysis. A design may be manufacturable under one set of shop floor resources, but not another. Likewise, the stock materials from which the part will be manufactured can impact the number of steps required, and hence the cost of the design. Further, the schedule of shop floor resources will have a significant impact on the process plan to be used and hence the cost of design.

Ideally, a product designer should be aware of all the technical constraints (materials, processes, tools, etc.) and shop floor constraints (routing, scheduling, etc.). But with the very large amount of information involved, a designer often needs the assistance of other domain experts. This process of information exchange, whether it be sequential or concurrent, can be very time consuming. One solution to this 'designer's dilemma' is having intelligent interacting entities which can supply domain-specific information to validate the product design, without any input from the designer.

The heterogeneous multi-agent architecture which has been developed at The University of Calgary involves

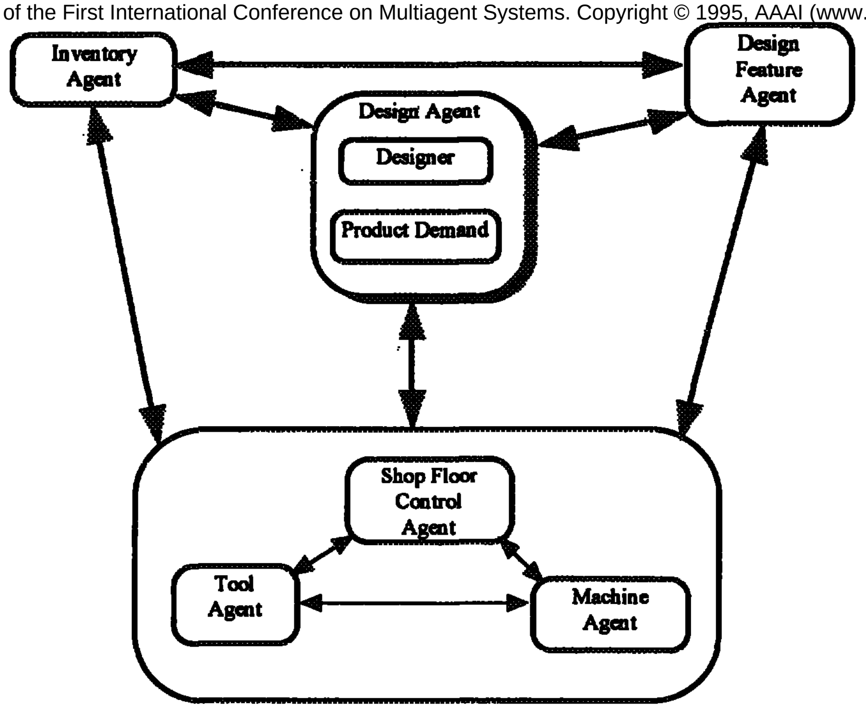

Figure 1 Design Interactions

intelligent interacting agents which collectively possess the necessary knowledge about manufacturability and shop floor control for concurrent product design and evaluation. These agents dynamically access, transfer, and evaluate much of the lower level design information, transparently to the designer. Figure 1 shows typical design interactions within this framework among the key agents involved.

In this architecture, the shop floor control agent imposes an 'adaptive hierarchy' over the shop floor resource agents and keeps track of the system state. The adaptive hierarchy reflects the ability of the shop floor control agent to maintain multiple virtual groupings of resources (machine, tool, material handling equipment, etc.), to accommodate the current management policies and system status. Such an adaptive hierarchy can help impose product priorities, changes in due dates and cope with internal disturbances (machine breakdown, tool breakages, etc.), by reassigning resource agents.

The design feature agents help to define and evaluate the local geometric, tolerance and other manufacturing specifications of the product, concurrently. Each type of feature has its own feature class, and each individual feature has its own feature agent. The concept of intelligent features was introduced by Mowchenko et al. [1994] in earlier work on multi-agent based intelligent design. The inventory agent helps in addressing concerns about material requirements for manufacturing the product. The machine and tool agents evaluate the manufacturability and produce process plans incrementally, as the design progresses. All of these agents are knowledgeable about their own domains and can advise other agents about violations or alternative solutions in their domain, upon request.

In this multi-agent design environment, the design agent carries out the design task oblivious to the constraints and restrictions of other related domains. Like any other agent, the design agent can request information from another agent about a violation or alternative solution in the latter domain. He or she can define variations and let the other agents decide on consequent outcomes. In all cases, the ideal is to arrive at multi-objective decision(s) concurrently, through the cooperative negotiation among different agents.

The architecture can be readily extended to incorporate other life cycle concerns such as assembly, service, recycling etc. by the addition of suitable agents. Thus the architecture is generic in nature. The ability to address design issues concurrently with the dynamics of shop-floor is a novelty with this approach.

The Agent Based Concurrent Design Environment (ABCDE) described below is a proof of concept prototype system implemented to demonstrate this multi-agent based concurrent engineering framework.

Implementation of ABCDE

The Agent Based Concurrent Design Environment has been implemented using an object oriented programming language (C++), which facilitated efficient coding through object oriented mechanisms such as parametrized types, multiple inheritance, and dynamic

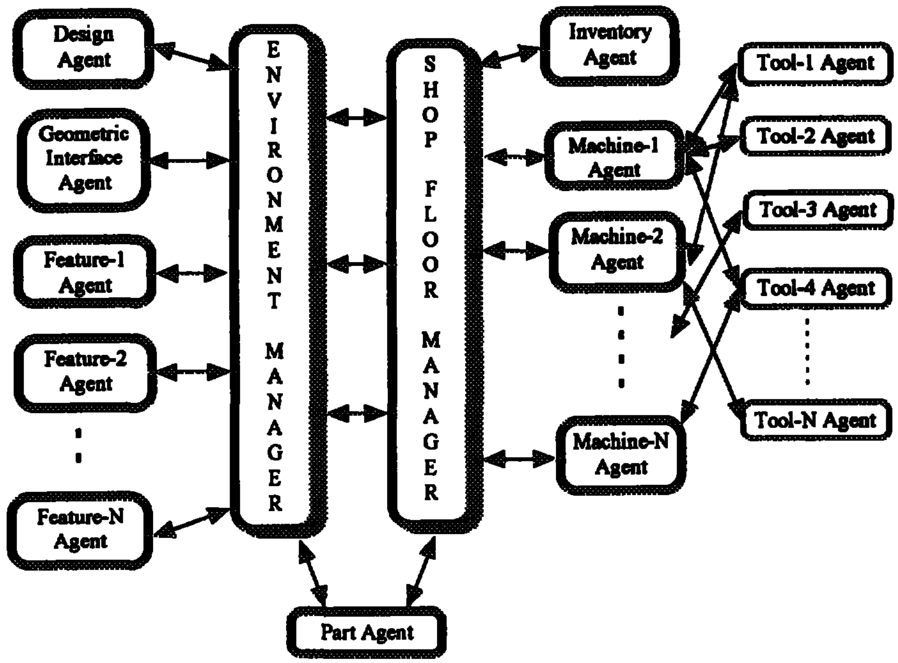

Fig. 2 ABCDE Agent Architecture

binding. The Advanced Modeling Extension and Application Programming Interface of Autodesk's AutoCAD were used to implement the geometric user front end, and for graphic display and manipulation. This was to provide a popular, low cost geometric front end, with readily available and proven drafting aids.

Being an agent based system, the organization of ABCDE is modular. The overall system architecture of ABCDE is shown in Figure 2. This multi-agent system is heterogeneous in nature involving the following different agent types: design agent, geometric interface agent, feature-type agents, inventory agent, machine agent, tool agent and part agent. The design agent is the human design expert. The geometric interface agent acts as a buffer to translate information between the geometric front end and other agents, and helps in separating low level application-specific coding.

As noted earlier, the feature types are for three-dimensional prismatic components, in this version. There is a feature agent class for every feature type. Imposing strict non-degenerative design features with a one-to-one relationship to manufacturing features could be very restrictive from the designer's perspective. Hence, the features are allowed to degenerate and the feature agents can detect degeneracy and resolve it appropriately. Consistency checking is carried out and resolution takes place, whenever a feature gets modified directly or indirectly. The major degenerate features types included are step, pocket, rectangular slot, U-slot, V-slot, dove tail slot, T-slot, flat bottom hole, taper bottom hole, counter sink, counter bore, wedge, thread, chamfer and fillet. There are a total of twenty-five non-degenerate feature types, internally. The feature agents evaluate local manufacturability when they are first created or whenever they get modified directly or indirectly. Local manufacturability evaluation includes accessibility, stability, and tolerance-surface finish compatibility, etc.

The inventory agent keeps tracks of the available stock materials and their shapes and advises about alternative stock sizes that could be used, if enough stock of requested size is not available. This helps the design agent in evaluating the stock-cost options and in materials requirements planning.

Every machine on the shop-floor is represented by an autonomous machine agent. This machine agent has knowledge about its machine's physical and process capabilities, probable tooling and schedule. Upon request, the machine agent explores the possibility of its machine contributing towards the manufacture of a feature and then evaluates alternative processes that can be used within its capabilities. If a satisfactory process is found, the machine agent auctions for tools relevant to the process and chooses one based on optimum performance-cost combination. If the machine agent cannot meet the tolerance specification, it will try to sub-contract the machining requirement. It also checks for global concerns such as accessibility, stability, and due date. Any anomaly or failure, unresolved or detected in the above process is promptly notified to the design agent.

Every tool in the shop-floor is represented by an autonomous tool agent. The tool agent has knowledge about its shape, schedule and tolerance capabilities in combination with a particular machine and work material, under certain operating parameters. This is helpful in automatic tool and operating parameter selection for a particular process request from a given machine. As was noted earlier, the selection process is accomplished through bidding by the interested machine agent.

The part agent dynamically updates and maintains itself as the repository for both product data and knowledge, as the design progresses. The product data includes the geometric and manufacturing specifications while the product knowledge includes the conditions under which a particular specification was accepted as manufacturable. This helps in consistency validation both when the conditions change and also when the product is to be modified later on in its life.

There are two agent coordinators, namely the environment manager and the shop-floor manager, and all of the other agents interact with each other through them. The presence of agent coordinators helps in eliminating unnecessary communication among agents which is irrelevant to the current task and in the dynamic restructuring of agent groups (e.g. upon machine/tool breakdown or addition). The environment manager merely acts as a message redirector and possesses no extra knowledge of its own.

The shop floor manager plays a vital part in shop floor control, in addition to its function and message redirection. It helps maintain virtual groupings of shop floor resources to accommodate current management policies and system status. It also helps in imposing hierarchy within a group as required. For example, it broadcasts about a feature requirement only to those machine agents that are in the highest level of the hierarchy (primary process machines, such as milling or drilling). These machine agents may then request the shop floor manager to sub-contract to machines at a lower level in the hierarchy (secondary or tertiary process machines, such as grinding, honing, or lapping). Apart from avoiding unnecessary communication among agents irrelevant for the present task, this also helps in the machine agents remaining independent of system status (i.e. they need not keep track of addition/deletion/regrouping of resources). The shop floor manager also resolves the bidding process, in the case of more than one bidder. It also notifies the design agent if either none or only incomplete specifications can be satisfied.

A primary concept in implementation was to make the system a design advisory aid and not to place any restrictions on conflict resolution. This means that the designer is not forced to clear up any or all unresolved issues before proceeding further, but if he does resolve all issues before proceeding so the product is guaranteed to be manufacturable. The manufacturing knowledge implemented in the system has been collected and synthesized from a wide variety of sources.

Being a proof of concept prototype application, the number of features and the degree of intelligence offered by the system is limited for a real world problem. However the architecture is modular and generic, and hence additional features can be incorporated over time.

A Design Session withABCDE

The designer can start the process by using a preprocessed blank geometry. It should be noted that this blank geometry may only be tentative and can be changed in the midst of the design process, if need arises. The inventory agent will automatically check the available stock sizes and shapes for the required quantity and advise the design agent accordingly. The design agent may evaluate available stock-cost options or may request the purchase department for stock procurement, or may postpone this discussion until later, in which case the process repeats at a later stage.

The design agent (designer) selects a feature for the part, which requests the environment manager to instantiate this particular feature type on the blank. The manager redirects the request to the appropriate feature agent. Next, the feature agent will request the design agent, through the manager, for dimensional, locational, tolerance and other manufacturing related information. After collecting enough information, it starts evaluating local manufacturability in terms of accessibility, stability, and so on. Violation of any constraint is promptly notified. If this agent has altered any existing feature, it informs the appropriate agent(s) to carry out consistency checking. Any unresolved issue is notified to the design agent by the concerned agent. All of the relationship information gathered in the process is stored for use during future alterations.

Subject to successful completion of local manufacturability evaluations, the feature agent requests the environment manager to evaluate global concerns. This request is redirected to shop floor manager who in turn auctions these requirements to machine agents belonging to the highest level of hierarchy in the different virtual groupings. As described earlier, these machine agents choose appropriate processes and tooling, evaluate global concerns and try to sub-contract for their shortcomings. This process repeats down to the lower level of hierarchy until either the specifications are met or the lowest level in the hierarchy has been reached. The later case means that the shop floor has insufficient resources to meet the requirements, reflected by nil bids for the feature. The success or failure of the bidding process is reported back by the shop floor manager to the design agent, back through environment manager and feature agent, for further action.

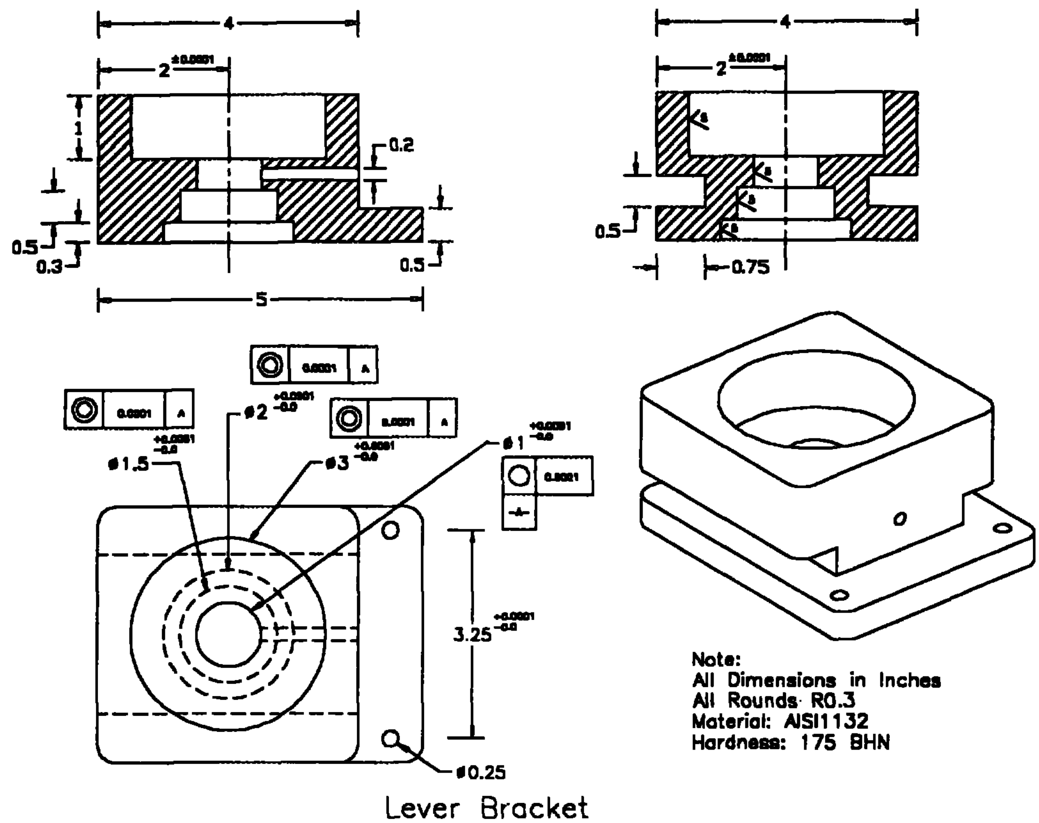

Figure 3

A successful feature thus far is accepted by the part agent as manufacturable. A failure triggers suggestions, if any are available, by the feature agent to the design agent. The final resolution is left to the design agent. The product is thus designed by repeated instantiation of features with concurrent manufacturability evaluations. Figure 3 shows a simple example of a product designed using the ABCDE system, involving only a few of the different types of features available. The system is undergoing further testing and evaluation.

Further Work

The alternative processes, operating parameters and tools chosen by the machine agents are components which would normally be included in the general process plans of the parts. These plans can be made specific by using these components with setup planning, operation sequencing and fine tuning of the operations. These specific plans can be combined with routing and scheduling through a cost based model and bidding process [13]. Production control for the entire demand quantity can then be achieved through a repeated bidding process. It should be noted that the process plans generated by the machine agents would remain the same but the routing and schedule would change according to the lowest bid. Work is in progress to integrate these aspects of manufacturing and shop floor control into the Agent Based Concurrent Design Environment system.

Conclusion

The volume of domain specific information involved and the complexity of addressing, integrating and automating various aspects of product life cycle requires new approaches to concurrent engineering design. A heterogeneous multi-agent framework for concurrent design provides a feasible solution for organizing design and manufacturing knowledge and integrating design with manufacturing and shop floor control, through the use of intelligent interacting entities capable of making multi-objective decisions, concurrently through cooperation and negotiation. The Agent Based Concurrent Design Environment implemented as a proof-of-concept prototype of a feature-based design system for prismatic components, amply illustrates the usefulness of this approach. This framework can readily incorporate other aspects of the product life cycle such as assembly or recycling as many be required. The modularity and flexibility offered by the multi-agent approach makes it an invaluable tool to automate and implement concurrent engineering practice.

References

Anderson, D.C.; and Chang, T.C. 1990. Geometric Reasoning in Feature Based Design and Process Planning. *Computers and Graphics*, 14(2):10-19.

Cutkosky, M.R.; Tenenbaum, J.M.; and Muller, D. 1988. Features in Process Based Design. In Proceedings of the ASME International Computers in Engineering Conference and Exhibition, 557-562. San Francisco, California.

Cutkosky, M.R.; Tenenbaum, J.M.; and Brown, D.R. 1992. Working with Multiple Representations in a Concurrent Design System. *Journal of Mechanical Design*, 114(3):515-524

Cutkosky, M.R. et al. 1993. PACT: An Experiment in Integrating Concurrent Engineering Systems. *Computer*, 25(1):28-37.

Dixon, J.R. 1988. Designing with Features: Building Manufacturing Knowledge into More Intelligent CAD Systems. In Proceedings of the Manufacturing International Conference, 1:51-57. Atlanta, Georgia.

Finger, S.; and Dixon, J.R. 1989a. A Review of Research in Mechanical Engineering Design. Part I: Descriptive, Priscriptive and Computer Based Models of Design Processes. *Research in Engineering Design*, 1(1):51-68.

Finger, S.; and Dixon, J.R. 1989b. A Review of Research in Mechanical Engineering Design. Part II: Representations, Analysis and Design for the Life Cycle. *Research in Engineering Design*, 1(2):121-137.

Genesereth, M.R. 1991. Designworld. In Proceedings of the IEEE Conf. on Robotics and Automation. 2785-2788. Los Alamos, California.

Gu, P.; Balasubramanian, S.; and Norrie, D.H. 1994. Bidding-Based Process Planning and Scheduling in a Multi-Agent System. Forthcoming.

Ishii, K. 1990. The Role of Computers in Simultaneous Engineering. In Proceedings of the ASME International Computers in Engineering Conference and Exhibition, 1:217-224.

Jackeila, M.; Papalambros, P.; and Ulsoy, A.G. 1985. Programming Optimal Suggestions in the Design Concept Phase: Applications to the Boothroyd Assembly Charts. *ASME Journal of Mechanisms, Transmissions and Automation in Design*, 107(2):285-291.

Jackson, S.D.; Sutton, J. C.; and Zorowski, C. F. 1993. Design for Assembly Using Fuzzy Sets. In Proceedings of the ASME National Design Engineering Conference, 117-122.

Kwok, A.D.; and Norrie, D.H. 1992. IAO: A Multi-Paradigm Programming Environment. In Proceedings of the International Conference on Object Oriented Manufacturing Systems, 219-226. Calgary, Canada.

Mahajan P.V.; Corrado Poli; Rosin, D.W.; and Wozny, M.J. 1993. Design for Stamping: A Feature Based Approach. In Proceedings of the ASME National Design Engineering Conference, 29-50.

Makino, A.; Barkan, P.; and Pfaff, R. 1989. Design for Serviceability. In Proceedings of the ASME Winter Annual Meeting, 117-120.

Marco, D.P.; Eubanks, C.F.; and Ishii, K. 1994. Compatibility Analysis of Product Design for Recyclability and Reuse. In Proceedings of the ASME International Computers in Engineering Conference and Exhibition, 105-112.

Mowchenko, M.; Norrie, D.H.; and Balasubramanian, S. 1994. Intelligent Independent Features: Manufacturing Features Which Ensure Their Own Manufacturability. Forthcoming.

Regli W.C.; Gupta, S.K.; and Nau, D.S. 1994. Feature Recognition for Manufacturability Analysis. In Proceedings of the ASME International Computers in Engineering Conference and Exhibition, 93-104.

Salomons, O.W.; Van Houten, F.J.A.M.; and Kals, H.J.J. 1993. Review of Research in Feature Based Design, *Journal of Manufacturing Systems*, 12(2):113-132.

Shah, J.J.; and Rogers, M.T. 1989. Feature Based Modelling Shell: Design and Implementation. In Proceedings of the ASME Computer and Engineering Conference, 255-269.

Shoham, Y. 1993. Agent-Oriented Programming. *Artificial Intelligence* 60:51-92.

Wright, T.L.; and Hannam, R.G. 1989. A Feature Based Design for Manufacture: CADCAM Package. *Computer Aided Engineering Journal*, 215-220.