PROCESS MANAGEMENT IN A DATA INTEGRATED DESIGN ENVIRONMENT

Tay-Sheng Jeng and Charles M. Eastman[+]

ABSTRACT: This paper presents an approach to supporting cooperative design processes in a data integrated design environment. We address the modeling of CAD-based design activities in a manner that captures domain specific knowledge, maintains consistency in process flows, and coordinates design tasks. Given this perspective, we propose a design framework by which global objectives can be tracked and managed, partial design results and isolated automation tools can be integrated, processes can be identified immediately and re-executed as needed corresponding to signals of inconsistencies in design data or with goals.

INTRODUCTION

With respect to increasing computerization of design activities and the involvement of multi-disciplinary agents in engineering and design, there are several emerging problems to be recognized in the area of computer-aided design. First, in order to facilitate design tasks and preclude designers from routine or low-level detail design, it is recognized that a growing population of sophisticated CAD tools are being developed and employed in the product development process. Pieces of design process can be automated by using a variety of heterogeneous CAD applications during the process. A control strategy is required for dealing with the sophisticated, yet complex and disordered CAD-based design activities in the course of design.

Second and more important, in terms of the growing complexity of design products generated by multiple agents, today's designers are facing an increasingly complex production process as well. Much effort needs to be devoted to maintain design integrity across the multiple views inevitably used by participants. Unfortunately, this effort goes beyond the capabilities of current CAD environments [Stark, 1992].

Based on these premises, it is highly desirable to be able to re-conceptualize engineering and design processes to meet the production and organizational needs for design automation. The requirement reveals the strong demand for a conceptual framework that provides an integrated design environment for exchange of design information, coordination of design activities, and, in particular, better support for controlling a diversity of CAD applications in the overall design process.

[+] Department of Architecture and Center for Design and Computation, University of California, Los Angeles
{tjeng,chuck}@gsaup.ucla.edu http://www.gsaup.ucla.edu/people/{tjeng,chuck}

In this paper, we present an approach to supporting cooperative design based on the notion of integration of information processing over the design life cycle. The major focus of our work is to develop a design framework by which global objectives can be tracked and managed, partial design results and isolated automation tools can be integrated, processes can be identified immediately and re-executed as needed corresponding to signals of inconsistencies in design data or with goals. Through the framework, the design activities can be efficiently managed during the course of design, resulting in a more reliable and systematic approach to designing.

There are several issues that we believe should be included in the core of a design framework. The following is a list of features that we consider important for managing cooperative design.

1. ***process representation***: What is the representation of design process? How does a design tool process design information in an information-sharing environment (e.g. check-in and check-out)? What are the relations among design processes and how can they be represented?
2. ***change propagation***: How do we propagate change from a CAD application to another? How can a design agent be notified to re-execute the tool corresponding to change? If inconstancy occurs, how do we repair the correctness?
3. ***task coordination***: How do we deal with constraint conflicts across views of participants? What is the framework of negotiation?
4. ***process planning***: Can we pre-define a process plan by predicting the behavior performed in the course of design? Can we develop an innovative design by evolving or reusing an existing design plan?
5. ***resource allocation***: How can we identify and relocate the resources availably in a design organization? How can a design plan incorporate an organizational policy?

The main questions we address in this paper are the issues of 1,2,3 and 4 above. The issue 5 involve organizational resources and policies, which is beyond the scope of this paper.

In the following sections, we develop a conceptual design framework to cope with the issues 1,2,3 and 4 above. In section 2, a taxonomy of cooperative design is provided by identifying three levels of abstraction. The taxonomic abstractions define the fundamental requirements for developing a conceptual design framework. In section 3, a data integrated design environment is introduced for cooperative design. In section 4, we provide several fundamental concepts with schematic definitions for process modeling, by which design processes and their relations are represented. Given the schema definitions, we propose a design framework consisting of three layers in section 5. The framework is subsequently elaborated with emphasis on the process level in section 6,7, and 8. An example drawn from a simplified version of building design databases is provided to demonstrate how a design is managed in our framework. Finally, a summary of the contributions of the design framework is made in the end. The current status of our work and desired future development are described.

A TAXONOMY OF COOPERATIVE DESIGN

The process of cooperative design can be characterized by a set of decentralized processes that execute in parallel and iteratively. Each stage of design involves intensive interactions across processes and coordination of process iteration. In order to reflect the inherent characteristics of cooperative design, this section provides a taxonomy with different levels of design abstractions, each of which embodies different semantics and requirements for process management. The abstraction levels of design processes are shown in Figure 1, consisting of *tool-based task execution, consistency-driven process flows,* and *goal-oriented task coordination.*

Tool-based task execution. The very bottom level of abstraction copes with the representation of design processes and repetitive task execution. There are two focuses in this level. The first is to determine the boundaries of CAD tasks. An overall design process can be partitioned into a coherent set of CAD tasks according to resource allocation, organizational requirements, availability of CAD tools, and more generally, the goals of product development. To logically approach process planning, a means to appropriately determine the boundaries of CAD tasks is necessary. The second focus is addressed on process iteration. Note that a design is incrementally achieved and often relies on iteration of the services of the CAD tools that apply expert knowledge to a specific problem domain. It is necessary to be able to identify design data, the involved operation instances, and the relevant integrity constraints used in the earlier stages of design, such that design rationale can be recovered.

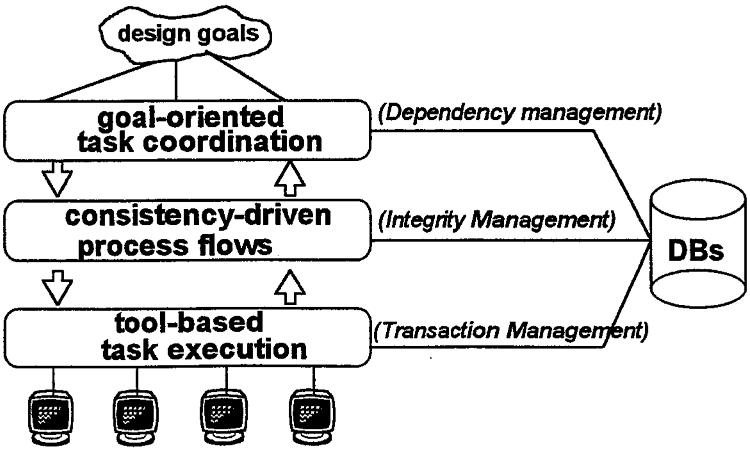

Figure 1. abstraction levels of design activities

Consistency-driven process flow. The second level on top of task execution copes with sharing data and interdependencies between CAD tasks. The major concern in this level is to maintain data consistency during the design process. Design data is highly dependent upon derivation output of task execution and is subject to change. To maintain consistency, change in one process must be propagated across multiple disciplinary processes.

Goal-oriented task coordination. At the highest level of abstraction, we deal with overall goals as well as design requirements with respect to the possible conflict of domain constraints across CAD tasks. Information modeling in this level requires a global view of the design system that aggregates information from the relevant CAD tasks for evaluation of a specific goal. To shorten design cycles towards a goal, we need a mechanism to simplify the coordination process to reach an agreement and to reduce semantic ambiguity of decision making for design alternatives.

A DATA INTEGRATED DESIGN ENVIRONMENT

It is generally believed that there is a great demand for integration in engineering and design, which requires a data integrated design environment for supporting information management over the design life cycle [Encarnacao and Lockemann, 1990]. In our opinion, a data integrated design environment has two aspects. On one hand, it provides a data repository through which design information is shared by multiple agents and design consistency is maintained. On the other hand, it serves as a process manager that controls a diversity of heterogeneous CAD applications and facilitates the coordination work during the design process.

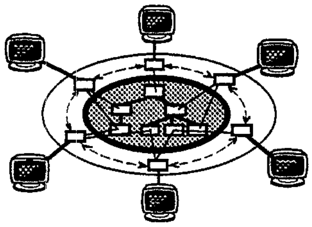

Figure 2. the architecture of a data integrated design environment

In this paper, we divide the system architecture of a design environment into three layers, as depicted in Figure 2. Central to the design environment is a kernel product model that integrates multiple views of design. The kernel is denoted by the shaded area with a hierarchy in Figure 2. We assume the kernel product model has the following capabilities:

- ability to support the interfaces with multiple heterogeneous applications, any of which can generate data as well as read it.
- identify inconsistencies and failures of logical integrity with the data as it evolves

- provide translation capabilities between the views associated with various applications.

Development of product models and databases with these capabilities are a priority at several research centers. At the outermost layer, there exists a set of real-world CAD applications being used for designing. We define this layer as an external layer of CAD applications in a design environment. Between the kernel and the external layer is a shell that conveys design formation from CAD applications to the kernel product model. The shell serves two purposes. To the kernel product model, it is an interface that manipulates design data and monitors design changes. To the external CAD applications, the shell plays an important role as the back-end knowledge carrier that manages process information and coordinates the concurrent design activities. In this paper, we refer to this shell as *a process model.* We wish to answer the questions raised in section 1 by defining the functionality of the process model within this architecture. The functionality of the process model is elaborated with several modeling concepts provided in the next section.

MODELING CONCEPTS

Several process modeling paradigms have provided constructs for representing and reasoning of a computational process with their specific objectives [Curtis, 1992]. In order to model real-world design information, this section introduces several fundamental concepts for representation of cooperative design processes. The modeling concepts lay the foundation for schematic representation of design processes and are necessary to explain our proposed design framework. The diagrammatic representation of conceptual primitives for process definitions are depicted in Figure 3.

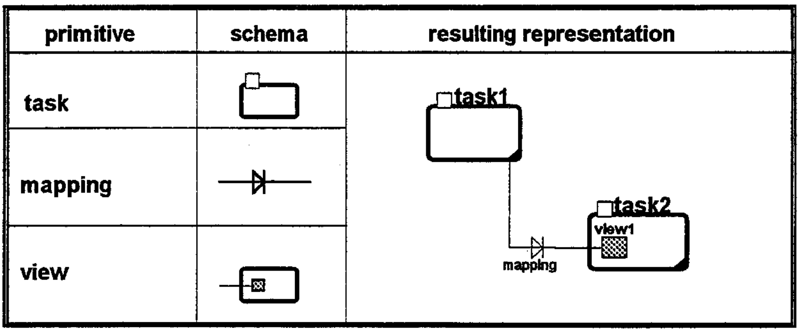

Figure 3. conceptual primitives for process definitions

Task. A task is conceptualized as the representation of a decentralized process that encapsulates the behavior of a CAD tool and semantic constraints for executing CAD applications. A design is transformed by executing the task repetitively, leading from the problem state to the goal state.

In this paper, we distinguish *a CAD task* from *a CAD tool* in three aspects. First, a CAD tool is a computer program embedded with domain-specific expert knowledge, in which some portion of the design problem is solved. On the other hand, a CAD task handles the service of a CAD tool without knowing how the problem is solved during tool execution. The CAD task concerns more how a CAD tool is to be invoked, how the arguments of a CAD task are derived in design context, and what are the pre and post conditions of the CAD task [Eastman, 1994]. Second, a CAD task is considered as an active design agent that retrieves input data, derives output data, and supplies CAD transactions to the central design databases. A designer is often part of a CAD task, but not a CAD tool. Third and more specifically, a CAD tool is encapsulated by a CAD task schema that serves as a back-end message carrier to the central database. The CAD task is considered as an integral part of our design framework, which maps the actual CAD tool to the schema of the central design database.

View. View is a subschema of a design product derived from production of task execution. In our work, we duplicated the shared data in multiple views to minimize the impact of changes in consistency maintenance. A task may contain a set of local views that define derivation inputs of the task, as depicted in Figure 3.

Mapping. Mapping is an active transformation by which multiple views are maintained with correctness and consistency. A mapping is comprised of a query and integrity rules. The mapping is an integral part of our process model for two reasons. On one hand, the mapping detects changes and maintains consistency between views by enforcing change propagation as changes are made to design. It holds data dependency between two CAD tasks. On the other hand, the mapping is specified as an internal application in back-end databases that bridges the gap between external CAD applications.

MetaTask. A meta task is a task that serves as an explicit basis for describing the logic or rationale in designing. Note that the design process is continuously evolving along with new CAD applications or innovative technologies used in the course of design. In order to capture the dynamics of design evolution, meta tasks identify variations or alternatives to a normative production process. The meta task consists of specifications of design goals shared by multiple disciplines and a global control structure for coordination of tasks.

A CONCEPTUAL DESIGN FRAMEWORK

We have presented the abstraction levels of design processes in Figure 1, which directly reflects the different requirements for developing a design framework. In his section, we propose a conceptual design framework that characterizes an integrated design environment and provides a mechanism to support cooperative design. The framework is a direct characterization of the architecture of an integrated design environment shown in Figure 2.

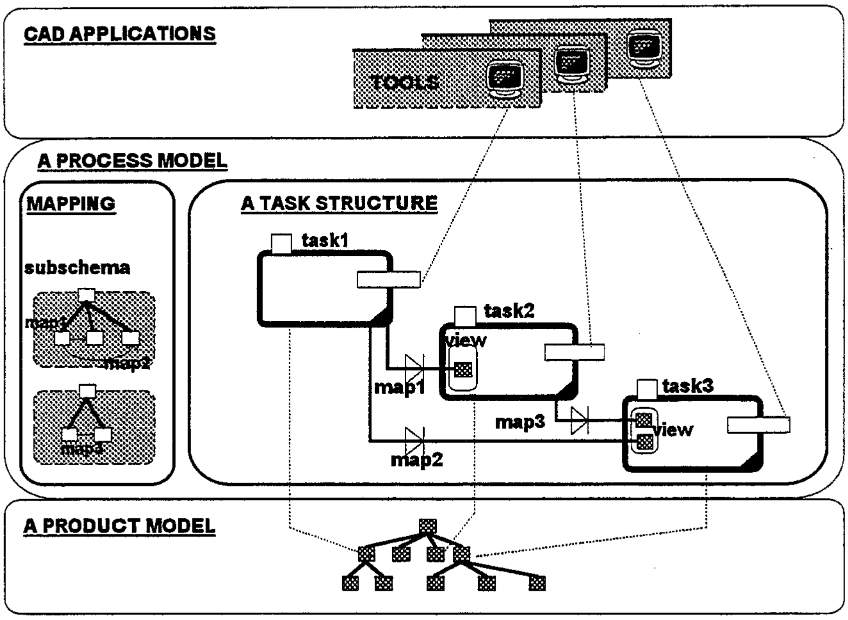

Figure 4. a conceptual design framework

The framework distinguishes three levels of abstraction: *external CAD applications, a process model,* and *a product model*, as shown in Figure 4. In this paper, we focus on the process level, consisting of *a task structure* for representing CAD activities and *mapping* that determines how semantic correctness of design can be maintained.

One can begin designing with one of the questions which is fundamental to our work: *Given a set of domain-specific design tools, how do we accomplish a design efficiently by making use of these CAD applications in an integrated design environment?* To a certain extent, this question can be answered by developing a design plan or a process flow embedded with tool procedures as well as control knowledge. In our design framework, designers construct a process flow by identifying design tasks and their relations in a process model. Each task is defined corresponding to the representation of the actual external CAD applications. In terms of the nature of cooperative design, the design data resulting from the task execution serves as the prerequisite for the execution of other tasks. A process flow is formed as a result of the information flow defined by the derivation inputs and outputs of each task. In the perspective of process planning, a process flow identifies precedence relations on the ordering of task execution, which can be as simply as a linear sequence, or may become a fairly complex network, encompassing parallelism and loops.

The repetitive task execution results in a coherent set of operation instances collected in the task schema. An operation instance resulting from task execution corresponds to a design transaction that derives pieces of a product model [Eastman and Kutay, 1989]. As the design proceeds, design transactions are represented and handled by operation instances in the task schema that convey design information between the external CAD application and the product model. In terms of our framework. This allows designers to trace design history by accessing pieces of design data and their involved operation instances. From another perspective, changes resulting from the repetitive task execution often lead to an inconsistent design. In order to maintain consistency, change propagation is carried out by the mappings that detect change in the product model and notify re-execution of the external CAD applications.

PROCESS MANAGEMENT FOR COOPERATIVE DESIGN

On the basis of the design framework, we develop a scenario for gaining an in-depth understanding of the architecture of an integrated design environment and suggest some ways to manage design more appropriately. An example drawn from a building design database with respect to three views of design, i.e. *spatial*, *structural* and *plumbing* systems, is provided to illustrate our design framework throughout the following sections.

Design Process Planning

One can start with a design by formulating CAD tasks in terms of the required CAD applications, and meta tasks dealing with the goals. For example, we can formulate a process scheme of building design according to the spatial, structural and plumbing systems that are assumed to be the major design tasks dominating the design process. Suppose our goal is simply to develop a sufficient interior space with structural and plumbing support. To represent the goal, we then introduce a meta task that aggregates information from the major tasks that is necessary for evaluation against the goal.

Due to the limited resources and design tools in design practice, there may exist gaps between the major design tasks. In order to bridge the gaps, we need to develop several internal CAD applications for use in the course of design. For example, based on these major tasks, we derive three internal mappings to maintain the dependencies between external CAD applications. The *map1* takes the coordinates of the grids and computes the span over two columns for use in the structural analysis. The *map2* and *map3* take structural elements and facility location for piping layout. With the specifications of the meta task, the external CAD tasks, and internal mappings, an initial process plan is then completed, as shown in Figure 6.

It is important to notice the distinction between a meta task and a process plan. In this paper, a meta task is considered as a prescription to achieve a goal. If one starts with formulating CAD tasks, the goals are left implicit and the logic or rationale has no explicit basis. Many tasks have simple generic goals, such as "*complete the design*", suggesting that normative processes exist for architectural design and many of its sub-tasks. Goals identify variations or alternatives to the normative processes. For a normative task process definition, we can refer to the professional architectural associations' codes of practice.

Note that our approach to process planning is different from traditional concerns that focuses on domain knowledge and expertise, fulfilled by design staff and outside consultants [Jacome, 1993]. In our approach, a second level is added, involving the task interfaces that are embedded in the flows among domain experts. We suggest process planning more than organizational resources here. To some extent, the configuration of a design plan involves change to the organizational policies and the discipline for managing projects. This may lead to radical change to the design world as a way of re-structuring design organizations for efficiency.

Tool-Based Task Execution

Given a design plan specified in Figure 6, an initial design is carried out by performing the design tasks sequentially. Let us start with the spatial grids laying out with (X,Y) coordinates. The distance between the grids is then computed by the span mapping (i.e. $S=X2-X1$). Based on the data derived from the mapping, the structural agent that takes the span and the load as parameters to determine the size of the columns (i.e. $Cx=f(S,L)$). Suppose that we have not considered the plumbing system yet. Then, the meta task computes the interior space by having the size of the column subtracted from the span (i.e. $N=S-Cx$). Finally, the interior space between two column is checked against the goal in the meta task.

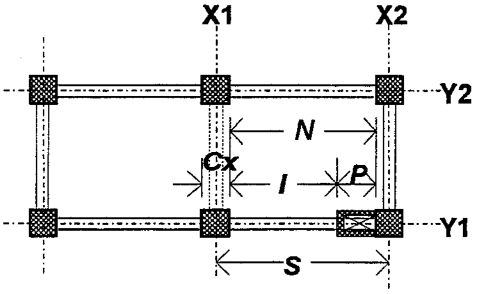

Figure 5. a simplified building plan

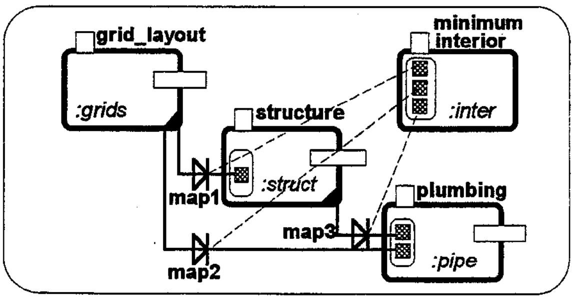

Figure 6 a process flow of building design

Normally, complete foresight of the implications of early decisions cannot be achieved and some tasks have to be re-executed to derive different results. This implies that the result of the first information flow, in this case *N*, may not meet all the design goals. In order to derive a satisfactory value, we need to rely on iteration of the services of the CAD tools that apply domain knowledge concurrently to reach the goals.

Consistency-Driven Process Flow

Design requires data consistency across multiple disciplines and depends on change propagation to maintain consistency in the process flow. There are two features of change propagation that should be noted. First, a consistent view of design is maintained by forward propagation that enforces change from output of a task to input of another. The change resulting from the upstream task execution often leads to re-execution of CAD tasks in the downstream of the process flow. For example, the spatial agent moves one of the grids (decreasing $X1$ by a value i), the span needs to be re-computed by increasing the size of S by i. Since the S is changed, the state of the structural task is no longer true. In the meantime, the next task handled by the structural agent is notified to re-execute the task, from which the dimension of the columns is re-computed. In our simple case, the change propagation involves deterministic functions, but normally they are not.

It is important to recognize that a design process continuously goes on evolving along with addition of new design applications or technological innovation to the process model. For example, we now consider adding the plumbing view to the building system. The process model is then expanded with addition of a piping task. Figure 5 depicts a piping core laying out next to the column. Apparently, the net space between columns is narrowed as a consequence of the layout of the piping core. In such a circumstance, we need to re-configure the task of net space computation and propagate change by replacing N with I.

Backward propagation is another situation we must consider in managing design information flow. Design often requires tracing back to the upstream CAD tasks due to inadequate or incompatible results generated from those tasks. It is important to notice the distinction between backward propagation and design cycle. When design tasks are deterministic, they often can be inverted, allowing backward propagation. While this is possible in the example, it usually is not the case, e.g., that designers can derive the maximum number of stories supported by a given target column size. When backward propagation is not possible, iteration and forward chaining or design cycle is required to resolve inconsistencies.

Goal-Oriented Task Coordination

We now consider the goal, i.e. designing a interior space with a minimum area *M*. Suppose that we end up with the value of *I* that is less than *M* after one round of task execution in the information flow. In order to eliminate the difference to reach the goal, we have three alternatives to enforce it: increasing *S*, decreasing *Cx*, or decreasing *P*. In other words, we need to take an action by either moving the grids, decreasing the size of the column, decreasing the size of the piping core, or even removing the column or pipe.

We identify two kinds of difficulties in goal enforcement and conflict resolution here. First, there may be a number of design methods and strategies to be taken to achieve a goal. However, due to the ambiguity of design results and unpredictability of domain knowledge, we cannot determine in advance what alternative we need to take. It is also not realistic to expect a pre-defined design strategy that always meets the specific needs in a problem context. For example, as depicted in Figure 5, the vertical pipe lines may conflict with the beams regarding the overlapped spatial artifacts. It would be inadequate to pre-define that piping has a preference for spatial preservation prior to beam layout. We suspect that rules for conflict resolution can be appropriately identified as variants for controlling coordination process in the meta task.

Second, the outcome of goal enforcement often results in recursive change propagation to achieve transitive closures. For example, if we expect to end up with the result of *(I=M)*, we may decide to move the grid of *X1* to left with an amount of *(M-I)*, such that the value of *I* is increased by *(M-I)*. As a result of increasing the value of I, however, the value of *Cx* may increase. We may end up with a new *I* that is still less than *M*. Most of the cases, we would either proceed with the refinement process (i.e. move the grids recursively) until evaluation satisfies the constraint, or roll back the update action performed by the grid_layout task and choose another alternatives.

CONCLUSIONS AND FUTURE WORK

In this paper, we have presented an approach to supporting cooperative design in terms of the notion of integration of information processing over the design life cycle. A conceptual design framework is proposed to provide an integrated design environment for exchange of design information, coordination of design activities, and, in particular, better support of controlling a diversity of CAD applications in the overall design process.

The design framework is developed based on a taxonomy of design processes that characterizes the nature of cooperative design in three levels of abstractions, each of which embodies different semantics and requirements for process management. In summary, some functional requirements identified here for the process level in the framework are:

- control of tool invocation and repetitive task execution,
- consistency maintenance in design information flows, and
- coordination of design activities for conflict resolution and goal enforcement.

We are currently considering several extensions of this work. First, a prototype system with the capabilities as described in the design framework is currently under development based on an object-oriented database system. Second, a language definition for representation of design process is proposed based on EDM product model concepts and EDM2 product database. Further investigation will deal with the expansion of the process level in the framework to include a patching model that minimizes process iteration, a coordination model that reduces ambiguity of decision making, and a mechanism for goal enforcement.

NOTE: This work has been funded by the National Science Foundation grant no. IRI. 9319982.

REFERENCES

Curtis, B., M.I. Kellner and J. Over [1992], "Process Modeling", *Communications of the ACM*, September, Vol. 35, No. 9, pp.75-90

Eastman, C.M. and A. Kutay [1989], "Transaction Management in Design Databases", *Computer-Aided Cooperative Product Development*, Proc. of MIT-JSME Workshop, Cambridge, November

Eastman, C.M. and G. Shirley [1994], "Management of Design Information Flows", *Management of Design: Engineering and Management Perspectives*, S. Dasu and C. Eastman (eds), Kluwer Academic Publishers

Encarnacao, J.L. and P.C. Lockemann (Eds). [1990], *Engineering Databases: Connecting Islands of Automation through Databases*, Springer-Verlag

Jacome, M.F. [1993], *Design Process Planning and Management for CAD Frameworks,* Ph.D. Thesis in Electrical and Computer Engineering, Carnegie Mellon Univ.

Stark, J. [1992], *Engineering Information Management Systems-Beyond CAD/CAM to Concurrent Engineering Support*, Van Nostrand Reinhold, N.Y.