Concepts for Computer Assisted Engineering Process Management

Jerzy W. Rozenblit
Dept. of Electrical and Computer Engineering
The University of Arizona
Tucson, AZ 85721-0104
USA

Christine Kocourek
Siemens AG
Corporate Research and Development
Otto-Hahn Ring 6
81739 Munich
Germany

Abstract

This paper presents concepts for computer aided support of engineering processes. We briefly discuss the basic tenets of the engineering enterprise and fundamental design and analysis techniques. Then, process modeling definitions are given and extended to encompass engineering process capture. An architecture is introduced that comprises executive, coordination, and execution layers, intended to provide an information system infrastructure for computer assisted support of engineering activities.

1. Introduction and motivation

The increasing global competition motivates major engineering companies to reduce the products' and systems' time-to-market and overall costs. At the same time improvements must be made in quality and efficiency of the product development and deployment processes. These improvements can be accomplished by providing an adequate methodological basis and, subsequently, computer-based systems for support of the life cycle of large scale engineering processes. Given such a basis and tools, it will be possible to better manage the immense complexity of the underlying process data, information, and knowledge.

The process of engineering, its formalization, and management have been the subject of research and development for the last few decades. Established engineering foundations and principles have been defined in well structured domains such as chemical or civil engineering. Subsequently, as processes and projects had become more complex, the discipline of systems engineering was formed [14,15]. Its primary concern is to ensure that all requirements for a system (that may comprise software, hardware, and human components) are satisfied throughout the complete life cycle [16]. As practiced today, systems engineering faces various methodological problems and errors. They include attempts to solve imprecisely stated problems, improper or inadequate use of existing technologies, budget and schedule overruns, etc. These are often due to the lack of an underlying engineering process methodology and personnel specifically trained to manage this process.

The other major factors contributing to the complexity of process engineering are: a) the multitude of principles, techniques, methods, and tools used in the engineering process and b) the emergence of new technologies that can be harnessed to support the management of such process (e.g. high speed computer networks, standard data exchange formats, multimedia, etc.). The technologies can facilitate a change in the work style and team coordination. This is demonstrated by the new paradigms of collaborative, distributed design techniques. Clearly, all those issues motivate us to examine how to devise methodologies and approaches that can harness the available technology to:

a) handle the complexity of large scale projects,
b) better support concurrent, collaborative, and distributed development of systems, and
c) automate or "semi-automate" routine engineering activities whenever and wherever possible.

2. Engineering enterprise

As defined by Blanchard [2], "engineering constitutes the systematic application of physical and natural resources combined in such a manner as to create, develop, manufacture and support a product or a process

0-8186-7889-5/97 $10.00 © 1997 IEEE

which economically provides some form of utility to man." This enterprise involves three basic components:

1. Body of scientific knowledge such as physical laws and principles.
2. Basic engineering comprising technical expertise and knowledge of economic, social, and political factors, individual skills such as analysis, modeling and simulation, ability to communicate and work with others, application of methods to solve problems, and personnel attitude characteristics such as objectivity and open-mindedness, initiative, willingness to develop and support of the product and process.
3. Product and Process - this is the engineering output providing utility and satisfying a need.

Clearly to accomplish the goal of an engineering process requires a team effort. This involves a combination of the following:

a) personnel to perform engineering functions
b) support personnel with technical skills that directly aid in the execution of engineering functions
c) non-technical personnel, e.g. accounting specialists, legal staff, etc.
d) physical and other resource required to accomplish process tasks and to construct the final product/system.

In the remainder of this paper, we briefly summarize traditional project design and analysis methodologies and present concepts for a generic organization architecture for computer-aided engineering process management. This architecture, which is intentionally domain independent, can serve as the basis for creating specific process support scenarios given an instance of a domain problem.

Before we proceed, we emphasize that our attention is focused on engineering as a dynamic process comprising a variety of activities rather than engineering as a body of scientific knowledge.

3. Project design and analysis methodologies: conventional views

Traditional approaches to represent, plan, and optimize project engineering. stem from methodologies that have emerged over the years in the operations research and computer science disciplines. They are unified by the underlying notion that projects can be represented in various ways as networks of activities. All of the techniques have a goal of generating a plan of project activities that will optimize a set of certain criteria (e.g. minimize cost or time, maximize the utilization or resources, etc.) More often, the optimality requirement is replaced by a problem of seeking satisfying solutions, especially in the presence of multiple criteria.

3.1 Activity networks

An activity network is a directed graph in which edges denote an elementary project step. The initial and the terminal vertex of an edge representing an activity "x" correspond to the events of "activity x starts" and "activity x terminates", respectively. Each activity has an associated duration expressed as a real positive number assigned to it. Vertices in the activity networks are synchronization points for the starts and completions of activities. As shown in [10], they can be interpreted as the AND/AND realization logic. In other words, a node is "realized" when all its input activities have been completed. At this point, its output activities can commence. This has the following consequences: it is neither possible to represent a choice in starting a single or some of the activities of a node nor to express that a certain activity will start at the termination of one or more out of several preceding activities. In general, designing a project based on activity networks is possible if the project specification meets the following requirements: [10]

a) a partial ordering of activities expressed as precedence in time is given,
b) synchronization of activities are consistent with the AND/AND realization logic, and
c) alternatives or choices are not required in generating the project plan.

The activity network language lends itself to the application of classical methodologies such as Critical Path Method (CPM) and Program Evaluation and Review Technique (PERT) [10]. Either technique allows the project managers to determine what is the least amount of time needed to complete the project (assuming no cycles in the network), and which activities are time critical and should be speeded up in order to reduce the total project time. This identifies critical paths and thus resources can be allocated on those paths to optimize the entire project. The difference between CPM and PERT lies in how we assign duration times to each activity. In CPM, activity durations are non-negative reals; in PERT we define them as non-negative beta-distributed random variables (we must be able to estimate the shortest, most likely, and longest durations). Again, both techniques are limited to a class of problems whose networks realize the AND/AND logic.

3.2 Petri-Net representations

Simple activity networks are not well suited for process modeling where monitoring and analysis of state changes is required. Petri nets are used for such applications. Many variants of Petri net specifications have been applied in well structured domains e.g. electronic design automation. They include condition-event, place-transition, and predicate-transition nets [11].

Bretschneider [4] developed a comprehensive theory and a supporting environment to demonstrate how Petri nets may be used to model engineering activities in VLSI design. In this formulation, predicates of a predicate-transition Petri net represent data or resources, design steps represent transitions, and tokens capture available resources or existing data objects. Using these definitions, Petri nets model concurrent and conflicting design activities. Bretschneider extends the Petri net model to include hierarchical processes, variable numbers of inputs and outputs, and knowledge-based decision-making with production rules. Although powerful in modeling the underlying processes, this representation remains complex. Designers that use a system based on Petri nets must first be well acquainted with the representation method. Furthermore, the complexity of Petri nets makes them inefficient for on-line support of design processes in real time. We have undertaken an effort to seek a simpler, yet expressive representation to model a process that can be mapped onto an equivalent Petri net [12,13]. This representation is now briefly summarized.

3.3 Hypergraphs

Our formulation is related to a classical view of how projects are represented by activity networks [10]. Initially, we have focused on the task flow graph (in the form of an activity on vertex, i.e. an AOV-network) in which no notion of alternative design steps could be expressed. In other words, implicit in the definition of the task was the assumption that all the steps must be carried out (in one of the possible orderings dictated by a precedence relation) to complete the project. However, in defining the task model, engineers might envision alternative activities that lead to the same result. To express this, we augmented the definition of an activity network with concepts adopted from the AND/OR graphs terminology [9].

An appropriate data structure to express a task flow graph is a *hypergraph* [9]. In a hypergraph, instead of edges connecting pairs of nodes, there are *hyperarcs* connecting a parent node with a set of successor nodes. These are called *k-connectors*, where k stands for the number of successors a hyperarc connects to a parent node. (Note that if all k-connectors are 1-connectors then we have a regular graph.)

Hypergraphs incorporate essential properties of a process. The AND nodes represent concurrencies, the OR structures represent alternatives, and the net structure reflects task precedence. The concurrencies incorporated in the AND nodes capture the essential non-linearity of design processes. Because alternatives are represented in the OR nodes, on-line planning and automation with hypergraphs can be changed quickly if an unexpected event occurs that requires changing the current plan. Furthermore, the precedence relationships of the hyperarcs can also be used like PERT charts to find critical paths and calculate project times. Hypergraphs can also be used to model processes with the information contained in data flow diagrams by showing the major activities and their dependencies. Like data flow diagrams, hypergraphs can show hierarchies of processes where each node of a high-level hypergraph can represent another hypergraph at a lower level. Most importantly, hypergraphs can be easily understood and manipulated by process engineers.

Hypergraphs facilitate human understanding and communication by modeling processes in a way that is easy to visualize and manipulate. A visual inspection of the net may also point to redundancies and bottlenecks in the process. Topological sort can be used to find a feasible linear ordering of activities and AO* search [9] can be used to find an optimal subgraph. Off-line planning and on-line planning are then possible to support process management and to automate all or some parts of the process.

4. Process modeling

In the early 1990's, a new research area has emerged called *process modeling.* Whereas this term has a strong connotation of traditional process modeling using network-like models (especially in the operations research community), it is a general designation of efforts to capture and describe software processes [7]. Research in software process modeling was motivated by several factors: a) traditional modeling techniques cannot adequately capture the multi-person, largely intellectual design activity with a high degree of cooperation, distribution, and coordination of various tasks, b) software products undergo continuous modifications and thus the development process is evolutionary in nature, c) design paradigms change rapidly as the underlying technology changes (e.g. object-oriented programming or the Java language). In summary, the most prominent characteristic that researches use to distinguish process modeling from other types of modeling in information systems, is the role of a human in the process. More specifically, many of the issues and aspects being modeled are enacted by a human not a machine. Thus the focus is not just on the user's behavior at the interface level but also on the inherent behavior of humans involved in the software development process [1,7].

Process modeling extends beyond software. It applies to business process reengineering, coordination technology where the aim is to manage dependencies among agents of a business process and to automate routine activities [7], and is well suited for large scale, complex engineering problems.

The concepts for software process modeling are rooted in the traditional definitions. A process is defined as a "partially ordered set of steps intended to reach a goal" [7]. A process step is described as an "atomic action." The constructs and concepts go beyond the operational definition of steps. They are augmented with the notions of:

a) agents, i.e. human or machine elements (actors, methods) that carry out process phases.
b) roles, i.e. a set of process elements assigned to an agent. A role is a unit of functional responsibility.
c) artifacts, i.e. an object, product created as a result of enacting a process phase, element.

In essence, a process model is to capture an interplay of agents who are assigned various roles and who interact to carry out process steps. Their actions are intended to produce artifacts consistent with the process requirements and specifications.

Curtis et al. [7] formulate a detailed set of objectives for software process modeling. We believe that those are important desiderata for engineering process modeling as well. Therefore, we summarize them below.

Desiderata for process modeling

Four fundamental objectives and goals of process modeling are:

a) To facilitate human understanding and communication, i.e. to provide a unified representation that is understandable to agents, that facilitates communication between them, and is sufficient to carry out the process.
b) To support process improvement by facilitating the reuse of its components, supporting a managed evolution of the process, and assisting in the selection and application of adequate, efficient technologies.
c) To support process management, i.e. to provide facilities to monitor, control, and coordinate the various elements, to provide planning and forecasting function, and a basis for process assessment.
d) To provide automated guidance in process performance and execution. This concerns both the support of cooperative work among the individuals and automation of the execution of formalized process components.

To accomplish the above desiderata, one must work in a methodological framework that integrates a multitude of various information forms and knowledge elements.

5. Beyond software engineering - process modeling at large

Discussing process modeling only in the context of software development is clearly too limiting. It is important to emphasize that various engineering, business and other domains (e.g. military) face similar and often more diverse issues. The human element that software process modelers use as a distinguishing problem characteristics is prevalent in engineering processes as well. Recall that an engineering enterprise involves technical, non-technical, and support personnel that must be assigned adequate roles, tasks, responsibilities and be appropriately coordinated to accomplish the process goals and objectives.

Historically, as we have pointed out in Section 4, modeling has been used extensively to manage and optimize flow of activities using network based approaches. Beyond flow optimization techniques, a suite of methods had been developed for systems analysis and design using operations research techniques. These mathematically grounded approaches, e.g. optimization and control theory, became more viable with the availability of powerful computational platforms that can solve complex problems using numerical approaches. Such methods, e.g. linear programming, are generic in nature and lend themselves to a wide variety of engineering and business applications. However, they require that the problem be well structured and convertible to the

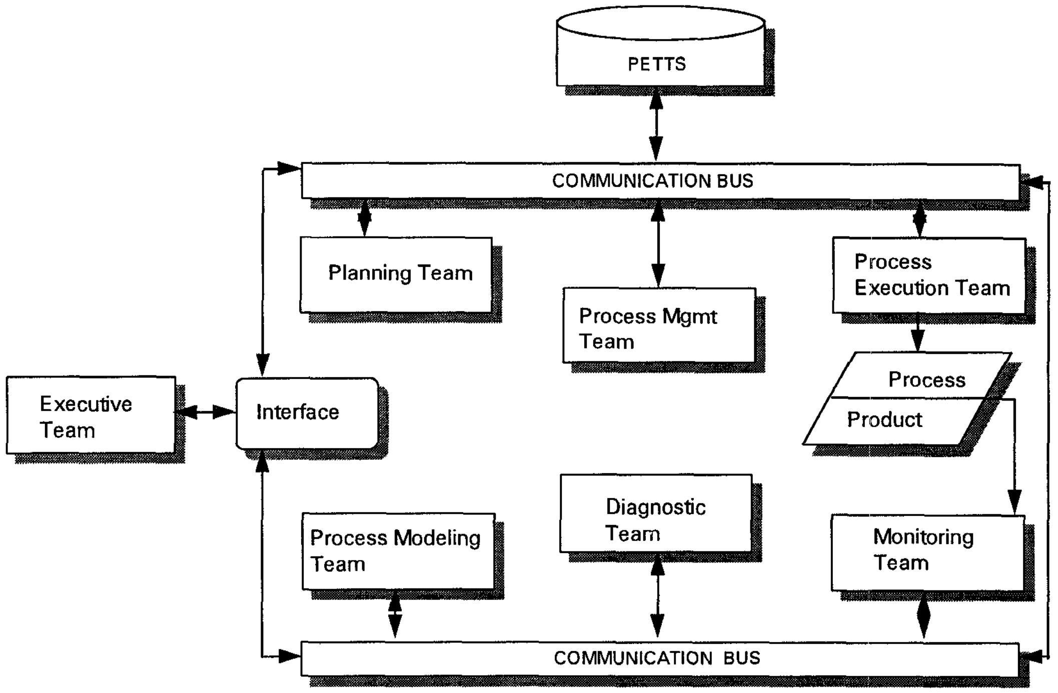

Figure 1. CAEPM Organization

underlying formal solution technique. We perceive these methods as a foundation for solving various problems during the entire engineering process and consider them an indispensable element of the engineering process toolset.

In addition to traditional areas such as civil, aerospace, chemical engineering, etc., where flow management and optimization methods have been widely used, manufacturing has become a showcase application of computer-aided process planning (CAPP) techniques. CAPP or automated process planning uses computer-based methods to generate a process plan, i.e. detailed operation instructions that transform an engineering design into a final part in a manufacturing facility [7]. The plan contains the route, processes, process parameters, machines and tools that are needed for production. The process planning functions involve the selection of machining operations, their sequencing, selection of cutting tools, calculations of cutting parameters, tool path planning, and generation of numerically controlled part programs. The process planner in CAPP must model part requirements, machines, tools, the interactions between parts, quality, cost, and other requirements. In recent years, many CAPP techniques have been proposed and are in widespread use in computer integrated manufacturing [7].

Among the new, emerging disciplines where process modeling appears to be given much attention, are: a) business process reengineering, b) engineering of complex, computer-based systems, and c) collaborative work environments.

Next, we propose a generic architecture intended to integrate various aspects of process engineering, its representations and modeling techniques.

6. Towards a computer assisted engineering process management organization

Our approach extends the previously developed theory and concepts for support of engineering design [5]. We now cast those concepts in the context of process engineering. The organization shown in Figure 1 is a scheme *aimed to unify teams (personnel), tools, methods, activities, roles, and knowledge that comprise an engineering process*. It provides a reference for an integrative environment in which projects can be carried out by teams with a high degree of computer assistance. We regard it as an information, human resources, and tool management system.

The CAEPM Organization is based on the concepts for the engineering design decision support architecture developed in cooperation with Siemens AG personnel [5]. It is viewed as an open software architecture with the following basic elements: a) Executive Layer, b) Coordination Layer, and c) Process Execution Layer. In addition to the layers, there is a central module called Process Engineering Technologies and Tools Set (PETTS). All modules interact with one another through communication interfaces. A man-machine interface is represented by the Interface block. We now describe these elements in more detail.

6.1 Executive layer

This layer encompasses corporate policy makers, system designers, and customers. We enumerate the basic decision making functions at this layer.

a) Corporate: decisions and trade-offs regarding the scope and viability of the proposed, new, and existing projects are made by the company management. Such decisions involve analysis of the overall company goals and objectives, market conditions as well as data from past and current related projects.
b) Engineering Process: given high level project goals and tasks, engineers refine such descriptions into project task specifications.
c) Management Interventions: monitoring overall project progress and interventions into the execution - an intervention may be called for by failures or may be stipulated by corporate decision makers and users, project engineers (for example, new goals or directions may be undertaken due to market conditions, etc.).

6.2 Coordination layer

The Coordination layer supports process planning, decision support, diagnostics, process flow management, and scheduling. These functions are salient to achieving high level of process support. The components of this layer are the Planning, Modeling, Process Management, and Diagnostic Teams.

Planning Team: This element is responsible for the generation of a process plan, i.e. a sequence of activities that can accomplish a project as defined at the Executive layer. An execution of a plan involves scheduling, i.e. a generation of time boundaries during which design actions should be executed and resource assignments required by those actions should be made. Scheduling should be the function of the Process Management Team.

Process Management Team (PMT): an element responsible for process control tasks. It acts as a process control flow supervisor and operates according to a process plan. PMT imposes a schedule for the Process Execution Team. Its functionality can be viewed at several levels:

1. Flow Execution Functions: activation of the process, selection of an activity (e.g. a tool or an estimator for it), scheduling, management of hierarchical process tasks actions, gathering process history information and process trace formation, backtracking to process flow alternatives.
2. Performance Monitoring and Evaluation: This function is to track and assess process performance. It is supported by information obtained from the Monitoring Team and process history.
3. Crisis Management: a capability of dealing with anomalies or failures in process execution. Minor anomalies reported by the Monitoring Team may be corrected by PMT through a choice of alternative methods, team or tool. Major failures may require replanning.
4. Learning: a function employed to improve the execution of process flow plans. Learning will lead to improved crisis management capabilities and optimized overall process.
5. Optimization: the application of operations research and heuristic methods to best achieve the overall goals, objectives, constraints, and performance requirements determined at the Executive layer.

Process Modeling Team: an element responsible for simulation modeling functions in the process. Simulation executes a process model according to the nominal or current (revised) plan. Model-based expectations are compared with the actual outcomes observed by the Monitoring Team. If the expectations and the actual process states match (with respect to the agreed upon criteria), the system progresses to the next action in its plan. When an expectation is not met, the system enters a diagnostic mode as explained below.

Diagnoser: This component identifies the causes underlying major anomalies in the process execution. Such anomalies may exist at two levels: a) between the virtual design process flow state and the actual execution state, and b) between expected specifications of a product and its actual state in the current process phase.

6.3 Execution layer

This layer carries out process actions. It also monitors the actual state of process and its resulting product. It consists of the Process Execution Team (PXT) and the Monitoring Team (MT). The PXT has members, each having a role, who use tools to work on activities by carrying them out according to methods. (Please recall the project class hierarchy detailed in Figure 3.)
The MT observes the process and product states. It compares the actual state with the expected state generated by the Process Modeling Team. The information collected by the Monitor is also used for performance monitoring, capabilities assessment, scheduling, actual flow trace recording, and display of process status information.

6.4 Process engineering techniques and tools set

This is a collection of tools and procedures salient to the management and computer aided process assistance. We conceive this element to be an open system that is populated by models, data bases, knowledge, standards, methods, and other components that may be required to carry out a process in a domain. Such components may include:

1. Process Model Base: a set of generic process models for a project domain.
2. Process Tool Base.
3. Experimental Frame Base: generic simulation experiment templates used in the evaluation of process models when simulation engines are employed.
4. Optimization Methods: set of mathematical optimization procedures for performance and tradeoff analysis.
5. Process Methods of project guidelines, rules, standard procedures for a given domain.
6. Technology Database: represents off-the-shelf components, standards, fixed parameters.
7. Process History Repository: record of previous projects in a given domain.
8. Standards
9. Representations of Work for Process Teams.

6.5 Functional modes

We have deliberately avoided making a distinction between human versus machine elements in the CAEPM organization. These elements, depending on the domain can be either, i.e. software agents or human team members carrying out specific process activities. We envision three degrees of system's functionality. Each carries a different level of components interactions. These functionalities are:

1. *Basic*: in this mode, CAEPM organization would support the engineering process management through off-line process modeling and selection of execution plans that best meet project's criteria. The plan generated by the Planning Team would be the basis for Process Management (PMT) and execution by the Process Execution Team. The Monitoring Team would report the resulting process performance and product characteristics to the Executive Team.
2. *Intermediate*: this mode would include on-line monitoring and cross checking with the nominal process model and process plan modifications. This would involve the communication among the monitors, diagnosers and planners in order to identify and rectify any problems or failures.
3. *Advanced*: an automated mode with a strong reliance on highly autonomous software agents. This mode would assume that processes are carried out semi-automatically with a shift of human roles to the executive level decision making.

7. Conclusions

We believe that the proposed CAEPM organization can meet the following set of process management desiderata [7]:

1. permit coexistence of both formal, informal, incomplete, and ambiguous specifications for various process elements,
2. support the binding of the execution of different parts of a process to both computing elements (automated tools) and humans,
3. support modifications during enactment,
4. represent both technical and non-technical aspects of the process,
5. provide analysis tools for process and product assessment,
6. provide planning, scheduling and simulation facilities and,
7. provide failure analysis tools.

References

[1] P. Armenise. A Survey and Assessment of Software Process Representations Formalisms. *International Journal of Software Engineering and Knowledge Engineering*, 3(3), 401-426, 1993.

[2] B. Blanchard. *Engineering Organization and Management.* Prentice Hall, 1976.

[3] P. Bradley et al. Business Process Reengineering (BPR) - A Study of the Software Tools Currently Available. *Computers in Industry*, 25, 309-330, 1995.

[4] F. Bretschneider. *A Process Model for Design Flow Management and Planning*, PhD Thesis, University Kaiserslautern. VDI Verlag, Reihe 9, No. 157, 1993.

[5] F. Bretschneider, C. Kocourek, S. Mittrach, and J.W. Rozenblit. Decision Support, Planning, and Data Management of Complex, *Discrete Event Systems. Proceedings of Artificial Intelligence, Simulation, and Planning in High Autonomy Systems*. IEEE Computer Society Press, 511-554, 1993.

[6] T.C. Chang. *Expert Process Planning for Manufacturing*, Addison-Wesley, 1990.

[7] B. Curtis, M. Kellner, and J. Over, Process Modeling. *Communications of the ACM*, 35(9), 75-90, 1992.

[8] E. Horowitz and S. Sahni. *Fundamentals of Data Structures*. Computer Science Press. 1976.

[9] R. Maier. First Transparency, Then Improvement. *Proceedings of the 1995 Intl. Symposium and Workshop on Systems Engineering of Computer Based Systems*, 211-219, Tucson, AZ, March 1995.

[10] N. Nilsson. *Principles of Artificial Intelligence*. Tioga Press. 1980.

[11] A. Pagnoni. *Project Engineering*. Springer-Verlag, 1990.

[12] W. Reisig. *Petri Nets, An Introduction*. Springer-Verlag. 1985.

[13] J.W. Rozenblit, L. Suantak. and F. Bretschneider. Concepts for Computer Aided Support of Complex Engineering Projects. *Proceedings of the 1996 European Meeting on Cybernetics and Systems Research*, Vienna, April 1996.

[14] L. Suantak, J.W. Rozenblit and C. Kocourek. A Hypergraph Approach to Design Flow Management. *Proceedings of the Sixth Annual Conference on AI, Simulation and Planning in High Autonomy Systems*, 79-84, La Jolla, CA, March 23-27, 1996.

[15] W.A. Wymore. *A Mathematical Theory of Systems Engineering - The Elements*, John Wiley and Sons, 1967.

[16] W.A. Wymore. *Systems Engineering Methodology for Interdisciplinary Teams*, 1976.

[17] W.A. *Wymore. Model-Based Systems Engineering*. CRC Press, 1993.