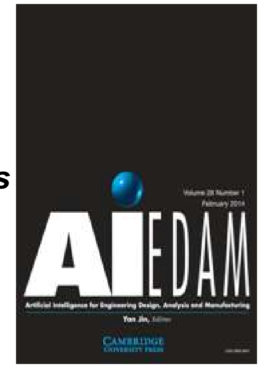
1-1-1994

On the Evolution of CAE Research

Clive L. Dym
Harvey Mudd College

Raymond E. Levitt
Stanford University



On the evolution of CAE research

Clive L. Dym and Raymond E. Levitt

Artificial Intelligence for Engineering, Design, Analysis and Manufacturing / Volume 8 / Issue 04 / September 1994, pp 275 - 282
DOI: 10.1017/S0890060400000950, Published online: 27 February 2009

Artificial Intelligence for Engineering Design, Analysis and Manufacturing (1994), *8*, 275-282. Printed in the USA.

On the evolution of CAE research[1]

CLIVE L. DYM[1] AND RAYMOND E. LEVITT[2]
[1]Department of Engineering, Harvey Mudd College, Claremont, CA 91711-5990.
[2]Department of Civil Engineering, Stanford University, Stanford, CA 94305.

(RECEIVED May 26, 1994; ACCEPTED June, 18, 1994)

Abstract

Less than a decade ago it seemed that a new paradigm of engineering – called computer-aided engineering (CAE) – was emerging. This emergence was driven in part by the success of computer support for the tasks of engineering analysis and in part by a new understanding of how computational ideas largely rooted in artificial intelligence (AI) could perhaps improve the practice of engineering, especially in the area of design synthesis. However, while this "revolution" has failed to take root or flourish as a separate discipline, it has spawned research that is very different from traditional engineering research. To the extent that such CAE research is different in style and paradigm, it must also be evaluated according to different metrics. Some of the metrics that can be used are suggested, and some of the evaluation issues that remain as open questions are pointed out.

Keywords: CAE Research; Computational Paradigms; Designers' Assistants

1. INTRODUCTION

It seemed in the early 1980s as if a new wave of computational power was about to sweep across – and radically change – the face of engineering research, education, and practice. The wave derived in part from the considerable success achieved in computational support for engineering analysis in domains as diverse as the finite-element analysis of structures and the nearly automatic layout of large-scale integrated circuits. The new computational wave also derived from and paralleled the success that artificial intelligence (AI) was expected to have in solving "real" problems that were not susceptible to numerical or algorithmic solution. The "hype" attached to the rapid growth of this branch of computer science was also felt in its application fields, and there was widespread optimism, at least among its adherents, that some AI paradigms – the most notable being knowledge-based (expert) systems – would provide extraordinary leverage for solving engineering problems. To be sure, there was (and still is) a countervailing view among traditional, engineering science-oriented researchers and practitioners that these emerging paradigms lacked the rigor needed to make a serious contribution, and history may prove that in some sense these skeptics are closer to the mark. However, at that time the pioneers of these new paradigms felt confident that their broader vision would emerge triumphant.

Along with the optimism and the hype, there developed the expectation that the emphasis of many engineering graduate programs would shift from an analytical, engineering science style to one that emphasized both numerical and nonnumerical computation to formalize a broad range of engineering tasks. The programs in *computer-aided engineering* (CAE) would stress software engineering, computer languages, programming environments, and a general understanding of software and hardware issues as the counterparts of traditional engineering knowledge in much the same way that many engineering science programs stressed applied mathematics as the essential implementation ingredient or "language" for doing any particular branch of engineering science. And it should be recognized that the thrust of such programs would be aimed both at engineering design *per se* and at synthesizing engineering design, analysis, and manufacturing. That is, the thrust assumed that computational tools to support engineering analysis were already accepted as an integral part of engineering practice. In fact, the success of these computational tools provided both a measure of credibility for the further application of computers in engineering and, as noted below, a need for still more powerful computational tools to support the tasks of design synthesis, documentation, assembly and manufacturing, and planning and control.

Reprint requests to: Prof. Clive Dym, Department of Engineering, Harvey Mudd College, Claremont, CA 91711-5990. Telephone: 909-621-8853, Fax: 909-621-8967, E-mail: clive_dym@hmc.edu.

[1]This paper is an adaptation and extension of Dym and Levitt (1994).

It was also anticipated that the growth of CAE would be fostered by a continuation of the decline in the cost of computing power. This cost trend would have the result of making workstations, high-end personal computers, and their associated computing environments readily and widely available. As a result of the increased availability of cheap computing power, graduate students and their advisers became advocates (and, in some cases, zealots) for the increased use of computers to support or automate engineering tasks. In addition, this cost trend influenced the development of techniques for distributing and coordinating complex engineering tasks. In particular, powerful local and wide-area networks linked engineers one to another and made it possible to decompose complex tasks organizationally and geographically. For example, large AEC firms such as Fluor, Bechtel, and CRSS routinely partition large design projects and use computer technology (e.g., different layers in shared three-dimensional CAD models) to allow teams of specialists to design different parts of the projects from geographically remote offices around the world. While such computer-aided project decomposition made it easier to employ distributed teams of specialists on complex projects, it also created significant new problems (and opportunities) in the areas of version control, data and information representation, database accessibility, and propagation of design standards and techniques.

The interest in the new AI-based paradigms and the declining cost of computing naturally fostered research into applications of these new styles of engineering computing. However, what developed less rapidly was a concomitant development of assessment and evaluation standards for these new areas of research and practice. Traditional engineering science research quite easily adopted the basic scientific approach wherein *reproducibility* was the key evaluation standard – that is, the ability of investigators to reproduce in their own laboratories the work of others done elsewhere. While developed primarily for experimental work, the reproducibility standard was also applicable to analytical work and, after suitable benchmarks were designated by the appropriate communities, to numerical computational work.

The situation for the AI-based paradigms did not allow a simple extrapolation of the reproducibility standard because many of the problems being encapsulated in AI-based programs were viewed as ill-structured – a viewpoint that is itself open to debate (Reich, 1994) – and their solutions as idiosyncratic because the expertise of particular domain experts was being used to develop new knowledge bases (a term, by the way, that did not even exist before the emergence of AI paradigms because numerical- and graphics-based programs were not thought of as repositories of "domain knowledge"). The application of the reproducibility standard was further complicated by widespread use of different programming languages and environments, and even more so by the different representation formalisms (e.g., rules and objects) as they became available.

Given the problems attendant to applying the standard of reproducibility, are there metrics that could be used to evaluate CAE research in general and AI-based research in particular? In the discussion that follows, four criteria for evaluating CAE research as an identifiable intellectual approach to engineering will be discussed and examined. They are as follows:

1. The routine use of KBESs and other AI-based programs in industry to do identifiable and meaningful tasks.
2. The ability of such programs, whether or not in practical use, to point toward useful ways of identifying and formalizing engineering knowledge.
3. The ability of such research to provide further understanding and clarification of the structure of engineering knowledge.
4. The ability of such research to extend and enlarge the engineering profession's understanding of engineering computation.

It is worth noting that these criteria are not easily quantifiable – that is, they cannot be subsumed within one or more "hard numbers." Thus, at least at this stage of development of the field of CAE, the community will have to accept qualitative assessments of the value of the ideas and techniques that are being developed and applied. In a sense, these criteria can be said to amount to a standard for assessing the *reproducibility of ideas*, rather than of specific answers or programs.

2. A HISTORICAL PERSPECTIVE

As noted, one of the forces pushing toward the increasing use of computers in engineering was the success achieved with analysis tools, most notably those numerical algorithms used to analyze complex designed artifacts such as aircraft structures, skyscrapers, integrated circuits, and sophisticated machinery. These algorithmic programs were based on procedural paradigms that employed numerical representations of various kinds of formulas. A related development, albeit one that came much later and rather more slowly, was the ability to draw and visualize devices of all kinds. These two developments made it possible for engineers and designers to consider, analyze, and evaluate a much broader range of design alternatives than were hitherto possible. That is, the cheapness of computer-based analysis allowed engineers to consider alternative configurations – especially in structural engineering – to a degree that simply wasn't possible when the extensive calculations needed to be done by hand, with slide rules, or with the aid of old-fashioned, mechanical calculating machines. The free-form, thin shells used in the Trans World Airlines (TWA) terminal at John F. Kennedy Airport in New York City became possible as alternatives to barrel vaults and hyperbolic paraboloids because of the availability of reliable computer-based analysis techniques. Other interesting structures—for example, the French headquarters of the International Innovation Institute (Texedeau & Souchet, 1991)—also became practical as structural analysts were freed from the need to rely on closed-form solutions.

In fact, the rapid spread of numerical algorithms in domains such as structural engineering has not only led to the consideration of more design alternatives—which might be considered a good result—but has also led to the increased use of analysis "black boxes" in rather mindless ways that waste both time and other resources. Sometimes, in fact, the mindless use of such tools is dangerous as well as wasteful, as in the recent instance of the failure of an oil platform under construction in the North Sea that was traceable in part to the inappropriate application of a finite-element code to analyze a complicated structure. Engineering educators see a microcosm of this problem on a daily basis when students deliver computer-derived solutions to problems: solutions whose physical dimensions and units are wrong, whose variables take on inappropriate values, and which show in other ways that students simply accept without question the results churned out by whatever software they are exercising.

In practice, of course, this meant that engineering analysis was being reduced to a technician-level job, wherein the practitioner was expected to know more about the ways to enter inputs than about the meaning and validity of the output. The ease with which computer-based analysis could be done thus produced both a burdensome multiplicity of needless, seemingly endless numerical analyses and a new bottleneck in the design process in which the engineering talent required to synthesize and evaluate meaningful solutions became increasingly less common than the talent available to generate analyses. The rapid spread of analysis tools was, of course, also fostered by the proliferation of cheap hardware and software.

Meanwhile, in academe and to a lesser extent in industry, researchers were experimenting with the new AI-based concepts—especially those falling under the rubric of knowledge-based or expert systems (KBESs)—to model various engineering tasks. Moving beyond the procedural paradigms and their numerical representations, researchers found that knowledge-based systems could be used to deploy engineering knowledge through declarative paradigms based on symbolic representation schemes (Dym & Levitt, 1991*a*). From the start, KBES technology was attractive because of its capacity to solve ill-structured problems requiring the formalization and application of heuristic knowledge, including that which is more scientifically based, even if not representable in formulas and algorithms. It was recognized that the rule-based paradigms which were the mainstays of early KBESs could represent deep knowledge compiled from first principles, as well as surface knowledge represented in associative heuristics. While such technology has begun to find a greater level of acceptance within the broader engineering community, it has required a relatively hard sell by those who saw its virtues early on.

As mentioned earlier, the contention that the problems and solutions encapsulated within such KBESs are ill-structured is itself open to debate (Reich, 1994). Indeed, since the knowledge being thus encapsulated is often deep knowledge based on well-understood physical principles whose application is also well known, one could argue that the problems are ill-structured only to the extent that they do not readily fit into the traditional differential equation or matrix algebra formalisms for the expression and application of such knowledge. In this view, it may be that the debate over the degree of "ill-structuredness" is not especially telling because the focus of discussion might more properly be on the understanding of the new formalisms and their limits, much as the understanding of the roles of mathematical, numerical, and graphical models are now well known. However, one related aspect of this is the depth of understanding that the developers of engineering KBES-based problem solvers have of the domain knowledge being encapsulated in these programs. There are several aspects of this issue that are worth exploring, including the achievements of AI researchers working on their own, the efficacy of collaborations between AI researchers and engineers and designers, and the domain backgrounds of newly-minted Ph.D.s in the infant discipline called CAE.

The first aspect is easy to deal with because there do not seem to be any success stories of engineering or design systems developed by AI researchers working entirely on their own. Indeed, once the developers of "smart" programs moved beyond solving classical puzzles (e.g., missionaries-and-cannibals, the 8-puzzle, etc.), they found that they needed to work with domain experts to achieve success in any field, whether the domain was medicine, geology, computer configuration, or engineering. This is a lesson that industry seems to have learned and absorbed, perhaps more so than the academic community. The most notable success stories, measured in terms of daily and consistent use of KBES technologies by companies that are interested in improving their efficiency and bottom-line performance, all involve active and deep collaboration between researchers versed in KBES ideas and techniques and respected domain experts. Three recent examples of this drawn from the authors' experience—there are certainly others—are as follows:

1. The PRIDE system for the configuration design of paper-handling subsystems of copiers was developed through a collaboration between a respected AI researcher, an engineer interested in exploring the representation of design knowledge, and a team of engineers and designers who were very experienced in

the design domain (Mittal et al., 1986). The PRIDE system is now in active use by designers at the Xerox Corporation.

2. The HAZTIMATOR system for configuring soil vapor extraction (SVE) treatment of contaminated soil was developed in a collaboration between researchers at Stanford University and hazardous waste remediation engineers at CM2 Hill (Oralkan et al., 1994). This system has been validated and will serve as a prototype for a full-blown commercial system.
3. The DEEP system for the configuration design of electric service for residential plats was developed by researchers at Harvey Mudd College collaborating with designers and planners at Southern California Edison (Demel et al., 1992). That system is currently being integrated into the day-to-day design environments used by SCE designers because SCE managers anticipate a substantial improvement in productivity as a result of this integration.

Thus, in terms of the first criterion of success mentioned above, it is clear that there are examples of success stories, and more are being developed everyday. It is worth asking, moreover, why these development projects produced successful results, whereas research projects undertaken by many academic researchers have not "taken off," that is, they have generally not led to some obvious real-world application or integration. In fact, it is interesting to note that, while these and other collaborative projects were being undertaken, a few advanced research programs were turning out Ph.D.s with an almost messianic zeal for the technology. This next generation of researchers should have formed a corps of shock troops who should have successfully propagated this new vision of engineering computing. But there was no explosion of graduate programs in the CAE arena. Although strong and viable groups were established at a few schools – the most notable one being that led by Professor Steven J. Fenves at Carnegie Mellon University (CMU) – the idea simply didn't take off. Indeed, while researchers in this area can now be found in many American engineering schools, it is rare indeed that there is more than one faculty member in a department whose principal focus is CAE. Indeed, even the Ph.D. recipients of the few programs that are active are finding it very difficult to find good academic positions, since they are often viewed as lacking sufficient depth in an engineering domain area (Fenves, 1993).

The lack of programmatic success was not due to a shortage of ideas or vision because, as researchers experimented with merging different computational paradigms and their representations [e.g., Levitt (1990)], an intellectual framework was also beginning to emerge, built on the notion that the AI-based work would serve as the "glue" for truly integrated computational environments for engineering [e.g., Dym & Levitt (1991*b*)]. There were early examples of such integration to lead the way [e.g., Darwiche et al. (1989) and Dym et al. (1988)]. Nor was the failure to propagate and spawn similar academic programs due to the expense of computing. After all, the cost of computer power was declining as, over time, KBES developers were able to migrate away from expensive Lisp machines and even more expensive KBES development software. In fact, commercial vendors began to produce KBES software that did not require a knowledge of Lisp or a deep understanding of underlying representation paradigms. These expert system shells were relatively easy to use, transparent, and portable to mid-level personal computers (such as Mac IIs and 286-based PCs). Thus, it appeared that all the ingredients were falling into place for the heralded computing revolution: cheap machines, software that was both more powerful and more "user friendly," highly visible research and graduate programs, a wave of graduate students who would spread the gospel to other institutions as new faculty members, and growing industry interest and support. Thus, CAE should have been a growth discipline in academia, and yet it has not turned out that way so far.

Further, in industry and in government, a parallel situation was unfolding in that the anticipated boom in KBES technology simply did not happen as expected (Feigenbaum, 1993). This in spite of incredible declines in the cost of workstations and software. In fact, even after expert system shells became relatively easy to use, transparent, and portable to high-end personal computers, there was no mass proliferation of KBESs that performed or assisted with engineering tasks which even remotely paralleled the proliferation of numerical analysis and graphics packages.

It is worth wondering, therefore, whether the failure of the academic discipline to stand on its own reflects – or could even be caused by – a lack of clear-cut, recognizable standards for evaluating success. Thus, it is worth examining what did not happen to see how the lack of rigorous standards for evaluating research may have influenced the course of development of CAE as a separate academic field.

3. WHY DIDN'T CAE(S) TAKE OFF?

Why did the anticipated boom never happen? One major reason is buried in a distinction drawn above, namely, that the real focus of these new approaches to CAE was design and design synthesis, what might be called CAE(S). That is, to the extent that experienced engineers feel that synthesis is the core part of what they do – the very "soul" of engineering – they are not willing to surrender it to a computer, at least not without a serious fight.

3.1. The lack of senior management "champions"

In industry, top management sees the value of reorganizing the work process to automate all routine work and, quite often, semi-custom work as well. Driven by the need to "re-engineer" their work processes in order to remain competitive, managers in large engineering organizations are more than willing to explore alternative ways to reduce the costs of doing their engineering work. Inasmuch as the kind of work is very labor intensive, the obvious managerial solution is to search for ways to reduce the number of engineers on staff. Thus, increasingly, managers are launching proof-of-concepts studies that focus on using KBESs that perform design tasks now done by experienced designers [e.g., Demel et al. (1992) and Mishelevich et al. (1991)]. While the engineers and designers provide the domain knowledge encapsulated in these KBES-based design systems, they also recognize that the success of such systems may make their own work redundant. Indeed, as a result, it may well be the case that engineers would viscerally oppose the thrust of this technology – as indeed all the books on labor relations say they should! In fact, the introduction of KBES applications has been quite successful only when senior management has been strongly committed to the idea (Riitahuhta, 1988).

In fact, notwithstanding the success stories presented in Section 1, there has been much more success in developing these approaches to design synthesis in Europe, with a concomitant increase in design efficiency and a consequent reduction in engineering staff. The story of the Intelligent Boiler Design System (IBDS) is instructive in this regard. Developed by Tampella Power Industries of Tempere, Finland (Dym & Levitt, 1991*a*; Riitahuhta, 1998), IBDS is a knowledge-based design system used for the semi-custom design of power generation boilers. Designs that would take months are now done in days, and Tampella's engineering staff now focuses on product research and development rather than on semi-custom design. Similar results can be reported for some of the design systems mentioned in Section 1, i.e., PRIDE and DEEP.

3.2. The KBES "consultant" paradigm

Another factor retarding the adoption of expert systems is the basic consultation paradigm employed by these systems. That is, rule-based KBES design systems are meant to be advisors or consultants, although in this context they provide advice *to* the engineers and designers. In this paradigm the engineer is then placed in the role of acting as a "sensor for an intelligent machine." Thus, such consultative systems are likely to be no more popular with engineering designers than earlier medical diagnostic KBESs were with physicians.

3.3. The fragmentation of the AEC industry

And, finally, from the industrial perspective, and especially in domains such as civil engineering and construction, the fabric of the industry is fragmented. Most AEC companies are in fact quite small, and they are labor intensive rather than capital intensive. Thus, they have limited resources for their own infrastructure development, and they have limited interest in sharing the development of appropriate tools with their competitors – unless they are forced into such investment by the competition. In addition, mitigating against cooperation and data sharing in such a fragmented industry are concerns about liability and about the ownership of intellectual property. Thus, industry as a whole has not been especially supportive of academic research in this area – which is in this case especially ironic because civil engineering departments are among the most active academic disciplines in CAE/CAE(S).

3.4. The lack of programmatic financial support

On the academic side, there may be several reasons that programs such as the ones at CMU and Stanford University have not flourished at many other places. One is, of course, that these ideas emerged at the same time as the beginning of a slide in the availability of external research support aimed at individual PI's and projects and while American colleges and universities were feeling sharp financial pressures. Thus, the inclination to invest heavily in new ideas and programs, which was never very strong in the rather conservative engineering education establishment, was further reduced by the lack of resources to hire new faculty and create new laboratories. The successful programs flourished because they obtained large-scale programmatic support, such as CMU's Engineering Design Research Center (EDRC) and Stanford's Center for Integrated Facility Engineering (CIFE).

3.5. The perceived lack of domain expertise of CAE researchers

Further, those departments that were willing to consider hiring new faculty in this area wondered about the depth of the engineering training that the new Ph.D.s were receiving in their engineering fundamentals. There was a concern that the depth of training in computer and software engineering was achieved at the expense of depth in technical areas such as structural or geotechnical engineering. Thus, prospective academic employers were somewhat skeptical that graduates of such programs would have the necessary engineering expertise. On the other hand, and somewhat ironically, for those industrial firms that wanted to experiment with KBES technology, such graduates were nearly ideal because they could talk both to in-house computing personnel and to the relevant engineers and designers.

The issue of the depth of domain knowledge of CAE researchers is also important in the context of research assessment and evaluation. It was noted above that the most successful outcomes—at least as measured by the first criterion identified in Section 1—were generated in active collaborations between academic researchers and domain experts from industry—and this dichotomy often overlaps with the distinction between computer scientists (the "academic researchers") and engineers (the "domain experts"). It is, in fact, part and parcel of the same problem. An essential ingredient of a successful collaboration, whether aimed at doing successful research or at building a useful product, is true collaboration between those who are really expert in the domain being modeled and those whose focus is the representation and programming technology. And, to the extent that academic "domain experts" lack real-world experience at dealing with seemingly ill-structured problems (i.e., problems not readily formulated in analytical or numerical terms) or that their training has emphasized the CA part of CAE, the prognosis for success is not good. That is, projects fail because the domain expertise is not sufficient to deal with the real problems, whatever they may be.

In saying this, one may in fact be repeating a criticism often voiced by engineers and managers in industry, that typical research-minded academics do not have a good grasp of real engineering problems. However, it is also often the case that academic "domain experts" are not especially aware of just how brittle their own expertise and knowledge is. The misuse of FEM packages, for example, is not limited to novice "black box" users in industry. In part this brittleness is a result of increased specialization, and in part it is likely a result of a significant diminution in the depth that doctoral candidates are expected to show before completing their academic training. This lack of depth will obviously be felt downstream as new faculty become responsible for training succeeding generations of researchers and practitioners. And, it must be noted, this problem is not limited to students in and graduates of CAE programs, wherein there is an explicit trade-off of depth for a certain kind of breadth. Rather, it is part of a general change in educational values and standards that is moving toward producing generalists.

4. LOOKING TOWARD THE FUTURE

What is to be learned from the past that will enable better preparation for the future, in terms of undergraduate and graduate education, research, and engineering practice? The main points are as follows. First, the age of integrated engineering computing environments is not far off. Graphics packages are already being converted from using layering for data storage and sorting to using object-based representations that facilitate the inclusion of object attributes and calculation methods with graphical and visual information (Dym, 1994). And while the KBES applications domain is not as large as anticipated a decade ago, there are still many researchers and practitioners who are developing KBESs for engineering design synthesis and as front ends to other engineering tools. And industry, in particular, is showing increasing interest in such integrated engineering environments in order to foster more intelligent design interactions that encompass all the phases of the life cycle of a designed artifact, their motivations being to increase both quality and efficiency (by eliminating apparently redundant staff).

Indeed, now that the cost of shells and their platforms has become so low, and their availability so high, many more companies are coming out of the recession of the last six years thinking about investing in this technology for the future. Higher level tools for design synthesis, based on AI concepts, are now available (e.g., Design++™, ICAD™). Such tools can be used to reduce both the cost of developing designs and the time it takes to introduce them. Further, as noted earlier, many design-intensive organizations—such as utilities and large AEC firms—are trying to reduce their costs by restructuring and shrinking their professional staffs. Thus, the combination of economic forces, the desire to reinvent the design organization, and the technical developments in the field may now lead to the long-delayed boom in the deployment of KBES design systems. And, as will be noted in the next paragraph, one of the consequences of the deployment of KBES technology will be some major shifts in the responsibilities and numbers of engineers who will be working in increasingly computer-intensive environments.

There are several implications for engineering education. The first is that as more engineering analysis and synthesis tasks are automated, there is less need for engineers who will simply run such programs as black boxes, oblivious to the built-in assumptions and ranges of validity of application. That is, the engineers will truly have to understand the physical fundamentals of their domains, as well as the expected behavior of the artifacts and devices they are analyzing and synthesizing. In design in particular, they will have to recognize the multiplicity of languages and representations of designed objects, so that they can be conversant with the ways their automated "designer's assistants" are manipulating synthesis information (Dym, 1993, 1994). Particularly at the undergraduate level, one consequence should be an end to the perennial debate about which programming language to teach engineering students. Except for a very small number of engineers who might actually be involved in writing software, most engineers will be users of increasingly sophisticated and accessible software. Thus, students should learn to become discerning and sophisticated users of packages (whether for spreadsheets, FEM, graphics, etc.) rather than amateur programmers.

At the graduate level, similar cautions would pertain. There ought to be renewed emphasis on fundamentals, on modeling, on behavior, and on the representations used to portray the different kinds of engineering knowledge that are applied in expert problem solving. While the need for continued research in AI-oriented representation and computational integration will persist, the emphasis will likely shift toward modeling issues that are more organizational than technical, as with current work on concurrent engineering and on modeling design teams (Levitt et al., 1994).

Note that these assessments of how undergraduate and graduate engineering education will change reflect, in effect, the success of CAE(S) as a field when measured against all four criteria set out in Section 1. That is to say, the changes in engineering education that are anticipated above mirror the application of new computational paradigms, a refinement of both the meaning and the structure of engineering knowledge, and continuing change in the roles that computers play in engineering practice.

Furthermore, it would not be surprising if the spread of automation to design synthesis for routine and semicustom work would lead to a significant reduction in the demand for engineering graduates, particularly those who stop their engineering education at the baccalaureate. An undergraduate engineering education is still perhaps the best approach to a "liberal education for a technological age," but it will no longer provide an immediate entrée to a well-paid professional life, as has been the case for the last five decades. Rather, the engineering baccalaureate will more likely become an indicator of educational achievement, much as current B.A. and B.S. degrees are viewed. Thus, the number and kinds of undergraduates in engineering might also shift dramatically in the future—and the numbers likely in a downward direction—with obvious consequences for engineering schools and colleges.

As a natural consequence of some of the previous possibilities, the roles of engineering schools and the nature of their faculties might well be pushed toward a professional model with the characteristics of medical and law schools. In such a model,the professional degrees would clearly be awarded only at the graduate level, and the faculty will become more diverse in the sense that seasoned practitioners, who may do little or no research, but who will have experience and contacts in the "real world," would become mainstays of engineering programs. As accomplished practitioners and designers become part of the faculty mix, the reward structure—especially the expectations underlying tenure reviews—will have to change, as will perhaps the view of which faculty are truly the elite.

Thus, finally, it would seem that the second and third criteria may be met by the field of CAE(S), if not necessarily by individual programs and their creators, because this reinvention of engineering education reflects a fundamental shift in the engineering paradigm. This shift will move the profession away from the analytically oriented, engineering science paradigm toward one in which knowledge is applied through many structures and representations, often through computational means. The encomiums and rewards will be earned by those who can decide *what* needs to be done to solve an engineering problem and *how* that solution should be implemented, and can then articulate *why* the approach taken is the correct one.

REFERENCES

Darwiche, A., Levitt, R.E., & Hayes-Roth, B. (1989). OARPLAN: Generating project plans by reasoning about objects, actions and resources. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 2(3)*, 169–181.

Demel, C.T., Dym, C.L., Summers, M.D., & Wong, C.S. (1992). DEEP: A Knowledge-Based (Expert) System for Electrical Plat Design. Technical Report, Southern California Edison—Harvey Mudd College Center for Excellence in Electrical Systems, December.

Dym, C.L. (1993). The role of symbolic representation in engineering design education. *IEEE Trans. Education 36(1)*, 187–193.

Dym, C.L. (1994). *Engineering Design: A Synthesis of Views*. Cambridge University Press, New York.

Dym, C.L., Henchey, R.P., Delis, E.A., & Gonick, S. (1988). A knowledge-based system for automated architectural code-checking. *Computer-Aided Design 20(3)*, 137–145.

Dym, C.L., & Levitt, R.E. (1991*a*). *Knowledge-Based Systems in Engineering*. McGraw-Hill, New York.

Dym, C.L., & Levitt, R.E. (1991*b*). Toward the integration of knowledge for engineering modeling and computation. *Engineering with Computers 7(4)*, 209–224.

Dym, C.L., & Levitt, R.E. (1994). Whatever happened to CAE? In *Bridging the Generations: Proc. Int. Workshop on the Future Directions of Computer-Aided Engineering*, (Rehak, D.R., Ed.). Carnegie Mellon University, Pittsburgh, Pennsylvania.

Feigenbaum, E.A. (1993). Tiger in a cage: The applications of knowledge-based systems. Keynote Lecture, *AAAI Symposium*, Washington, DC.

Fenves, S.J. (1993). Personal communication, June.

Levitt, R.E. (1990). Merging artificial intelligence with CAD: Intelligent, model-based engineering. Henry M. Shaw Memorial Lecture, North Carolina State University, Raleigh, North Carolina; reprinted as Working Paper No. 28, Center for Integrated Facility Engineering, Stanford University, Stanford, California.

Levitt, R.E., Cohen, G.P., Kunz, J.C., Nass, C.I., Christiansen, T., & Jin, Y. (1994). "The Virtual Design Team": Simulating how organization structure and information processing tools affect team performance. In *Computational Organization Theory*, (Carly, K. & Prietula, M., Eds.). Lawrence Erlbaum Associates, Hillsdale, New Jersey.

Mishelevich, D.J., Katajamaki, M., Karras, T., Lehtimaki, H., Axworthy, A., Riitahuhta, A., & Levitt, R.E. (1991). An open-architecture approach to knowledge-based CAD. In *Artificial Intelligence in Engineering Design*, (Tong, C. & Sriram, D., Eds.), Vol. III, Academic Press, New York.

Mittal, S., Dym, C.L., & Morjaria, M. (1986). PRIDE: An expert system for the design of paper handling systems. *Computer 19(7)*, 102–114.

Oralkan, G.A., Staudinger, J., Levitt, R.E., & Roberts, P.V. (1994). The HAZTIMATOR: Knowledge-based time and cost estimating of hazardous waste remediation projects. Working Paper No. 23, Center for Integrated Facility Engineering, Stanford University, Stanford, California.

Reich, Y. (1994). What is wrong with CAE and can it be fixed? In *Bridging the Generations: Proc. Int. Workshop on the Future Directions of Computer-Aided Engineering*, (Rehak, D.R., Ed.). Carnegie Mellon University, Pittsburgh, Pennsylvania.

Riitahuhta, A. (1988). Systematic engineering design and use of an expert system in boiler plant design. *Proc. ICED Int. Conf. Engineering Design*, Budapest, Hungary.

Texedeau, J., & Souchet, R. (1991). Innovative architecture realized using the finite element method. *Structural Engineering Rev. 3(3)*, 181–185.

Clive L. Dym is the Fletcher Jones Professor of Engineering Design at Harvey Mudd College. He has previously held appointments at the University of Massachusetts at Amherst (1977–1991), Bolt Beranek and Newman (1974–1977), State University of New York at Buffalo (1966–1969), and Carnegie Mellon University (1970–1974). He has also held visiting appointments at the Technion—Israel Institute of Technology (1971), the Institute for Sound and Vibration Research at the University of Southampton (1973), Stanford University and Xerox PARC (1983–1984), and Carnegie Mellon (1990). After completing the Ph.D. degree at Stanford University (1967), Dr. Dym did research on a variety of problems in applied mechanics and acoustics. Since the early 1980s his research has focused on the development of knowledge-based (expert) systems for engineering design and analysis. He has published more than 100 archival journal articles, miscellaneous papers, and technical reports. Dr. Dym has also edited two and written seven books, the latest being *Engineering Design: A Synthesis of Views*, published by Cambridge University Press (May 1994).

Raymond E. Levitt is Professor of Civil Engineering at Stanford University, where he also serves as Associate Director of the Center for Integrated Facility Engineering. After completing the Ph.D. degree at Stanford in 1975, he served on the faculty at MIT for five years before returning to Stanford. Dr. Levitt's early research employed organization theory and risk analysis to model decision making in facility engineering organizations. Since 1980 he has explored the use of artificial intelligence and knowledge-based systems as nonnumerical tools for site layout, design synthesis, project planning and control, and other facility engineering applications. Dr. Levitt's industrial experience includes design, estimating, and on-site management positions in marine and building construction, and numerous consulting assignments on organizational design, safety management, and expert system applications. He is also a director of Design Power, Inc., and serves as a scientific or technical advisor to several companies in the United States and Europe. Dr. Levitt is also the co-author, with Professor Clive Dym, of the book, *Knowledge-Based Systems in Engineering*, published in 1991 by McGraw-Hill.