Bo Dahlbom and Lars Mathiassen

The Future of Our Profession

There's more to being a good engineer than a high level of technical competence.

An organization like the ACM, with a variety of publications, newsletters, interest groups, and conferences, provides an important forum for discussing the nature and changes of the computing profession. Such discussions should of course go on continuously, but there will be particular periods of rapid social and technological change in which they are more necessary. A number of articles in *Communications* over the past few years [1, 3, 4, 10] suggest this is such a time.

The purpose of this article is to contribute to this discussion from a Scandinavian perspective. We begin by presenting a framework for understanding our profession and for discussing possible changes. The framework offers competing conceptions in terms of professional focus, approach, and role. We then use the framework in an examination of the current discussion within the ACM. The cited articles—in spite of having different motivation and purpose—all identify a need for professional changes in education, research, and practice. But, we argue, the changes suggested are insufficient to meet the demands placed on our profession by the ongoing changes in technology and society. As a consequence, we propose a more radical position on the future of our profession, with *technology in use* as its foundation, and we outline a new curriculum for the education of computer professionals.

Competing Conceptions

Discussions of professional identity tend to be internal affairs, dealing with details of professional competence. If we expand our field of vision a bit, we will often find that parallel discussions occur in slightly different terms in other professions. It should come as no surprise that the issue under discussion here, the question of our professional identity, is an example of a much more general question that has been a recurring topic of debate during the process of modernization. Engineers have played a leading role in that process, turning a society of farmers and craftsmen into an industrial, high-tech society. How they think of themselves and their roles, echoes the two major, competing views of that process: the mechanistic and the romantic. (See the sidebar "The Mechanistic and the Romantic Views.")

To put it bluntly: if your focus is on machines, and if you use the machine as a model for how you think of the world, society, and people, then you are a *mechanist*. If your focus is on people, on how they interact and change, on the processes of developing and using machines rather than on machines as such, then you are a *romantic*.

In our book *Computers in Context* [2], we analyze our profession using the mechanistic/romantic dichotomy as a dimension to compare and discuss different, and competing, conceptions of our professional identities. We have developed a conceptual framework distinguishing among three positions on what is the focus, approach, and role of a computer professional. Our framework is simple (perhaps deceptively so), and is summarized in Table 1 and is further elaborated in three subsequent sidebars on professional focus, approaches, and roles.

This framework is used in teaching throughout Scandinavia, both at engineering schools and at business schools. We have used it ourselves in courses for professionals at the computing centers of major Scandinavian companies, and we have seen how a framework like this can enlighten experienced professionals as to the identity of their professional roles.

A simple use of this framework is to distinguish the identities of three different professions within computing: *programmers* in the software industry are engineers who build things; *support personnel* in computer departments rely on an evolutionary approach to help people use computers; and management and trade union *consultants* focus on power when they use computer technology to change things. It is more interesting, however, to use the framework to question such simple labeling, crossing the column boundaries, perhaps arguing that engineers ought to take a more active interest in questions of power, and that consultants should know more about artifacts. This is how we will use the framework here.

Table 1. Competing conceptions of the computing profession

Slogan	I build things	I help people	I change things
Focus	Artifact	Culture	Power
Approach	Construction	Evolution	Intervention
Role	Engineer	Facilitator	Emancipator

In our analysis of the recent discussions within ACM, we read the articles using our framework as an organizing principle, trying to determine what each of the texts have to say about professional focus, approach, and role. Our findings are presented accordingly in the following three sections, para-

The Mechanistic and the Romantic Views

The *mechanistic* world view was developed in the 17th century by great philosopher-scientists like Descartes, Newton, and Leibniz. Turning the idea of the clockwork mechanism, the machine, into a powerful conception of the universe, they used this world view as a foundation for a new science of nature, a new definition of man, and a new political program. This world view played a central role in the Enlightenment, and thus in the modernization of Europe. It contained all the elements that we have come to associate with a scientific attitude: truth as the mapping of the world, the use of mathematics as an exact language of representation, the idea of methods, of formalization and rules for calculation and deduction, the view of thinking as the competent manipulation of symbols, a penchant for rational means-ends thinking, for planning, analysis, and final solutions. Applied to society, this world view gave us such powerful principles of organization as bureaucracy and liberal democracy.

The *romantic* world view grew out of a reaction against mechanistic thinking, and was formulated toward the end of the 18th century, primarily by German philosopher-artists like Herder and Hegel. This reaction made much of the difference between organisms and machines, wanting to defend nature and everything natural against machines and everything artificial. The romantic philosophers were not interested in taking the universe apart like a machine, in analyzing it into its smallest atoms. They wanted to contemplate, understand, interpret, feel, and see through the world to its meaning, as with a poem or a painting. If the mechanistic philosophers of the 17th and 18th centuries tended to think horizontally, mapping the causal sequences of controllable machines, the romantics developed a vertical way of looking at things. Where the mechanists saw structures and systems, the romantics saw processes and change. Changes are the expressions of hidden, bottled-up forces, of eruptions resulting from power struggles and the unleashing of mounting tension. The romantics wanted to change our world by changing our experience of it. The world we human beings experience, the world of phenomena, is construed by concepts that vary from culture to culture and from time to time.

phrasing the actual texts to support our interpretations. We then evaluate the positions taken in the articles using our framework as a basis for criticism, summarizing and paraphrasing to substantiate our points.

Professional Focus

The conventional view of the computing profession within academia is that it is a set of experts who make their livings by solving problems in hardware and software, yielding faster, cheaper, smaller, and more reliable information-processing systems. In his program for "Educating a New Engineer," Denning argues that this view no longer reflects the practice of computing.

Recording, processing, and communicating information have become an enduring concern of all human beings for effective coordination of work and action, and computer professionals are seen as the people who take care of other people's concerns in this domain. The computer professional is no longer an expert providing solutions to general information-processing problems. Instead, the computer professional has become an expert partner with clients in other domains [3].

This change in perceptions of the computing profession implies a change in our understanding of innovation, the raison d'être of the profession. We have always, according to Denning, placed a high value on innovation. But the traditional understanding is that innovation means introduction of a new computer-based artifact that makes a set of actions more efficient. A new understanding is emerging, according to which an innovation is a shift of the standard practices of a community to help them achieve their purposes more effectively [3].

Signs of similar changes in professional focus can be found in the other texts. Turner et al.—concerned with the design of computing curricula—claim that computer professionals must be able to anticipate the impact of introducing computer-based artifacts into a given environment, and they must understand the responsibility they will bear in doing so, as well as the possible consequences of failure. Students of computing need to understand the history of the discipline, they should appreciate the philosophical questions, technical problems, and aesthetic values that play important roles in the development of the discipline, and they should discuss serious questions about the social impact of computing [10].

Hartmanis et al.—assessing the scope and direction of computing research and education—believe the field and society will benefit if a broadening course is taken rather than if efforts at the core are increased. They suggest that members of the academic field of computing must intensify their intellectual interchange with other disciplines, focusing more on applications of computing in areas of economic, commercial, and social significance, and realizing that substantive research problems often emerge from such applications [4].

The "ACM Code of Professional Conduct" [1] exhorts computing professionals to give comprehensive and thorough evaluations of computer systems and their impacts, including analyses of possible risks. They should manage personnel and resources in such a way as to design and build information systems that enhance the quality of working life. Moreover, they should ensure that those who will be affected by a system have their needs clearly articulated during the assessment and design of require-

The Professional Focus of Computing

Artifact focus: The computing profession is concerned with technical, computer-based artifacts. The professional focus is on tools and techniques for development of such artifacts for individuals, organizations, and markets. Questions of quality address the artifacts themselves, and primarily their technical functionality.

Culture focus: The computing profession is concerned with computer-based artifacts in the practical context of their use. The professional focus is on adapting artifacts to the practice of individuals and the different cultures of organizations. Questions of quality concern quality in use, the way artifacts fit organizational contexts, the way they influence and are influenced by, individual practice and organizational culture.

Power focus: The computing profession is concerned with the role of computing in changing society and the lives of people. The professional focus is on moral and political issues related to when and how to use computer-based artifacts. Questions of quality concern the impact of artifacts on the distribution of power, autonomy, integrity, and democracy [2].

ments and that the system subsequently is validated to meet these requirements [1].

Analyzing these positions using our framework (see the sidebar "The Professional Focus of Computing") we see a change from a narrow, traditional artifact perspective to a broader focus in which the cultural context of computer-based artifacts plays an important role and in which moral and even political issues are included.

A recommendation to broaden the professional focus to include the cultural context is clearly expressed in all the writings: the computer professional should help clients in other domains to achieve their purposes more effectively through a shift of the practices of the communities in question [3]; computer professionals must be able to anticipate the impact of introducing computer-based artifacts into a given environment, and they must understand the responsibility they have in doing so, as well as the possible consequences of failure [10]; research efforts should be broadened rather than redoubled at the core [4]; and, finally, computer professionals should provide comprehensive evaluations of computer systems and their impacts, including analyses of possible risks [1].

The idea of focusing on power is more vaguely expressed, and with important differences between the writings. There are only a few and rather vague hints at a power focus in Denning and Hartmanis et al., while Turner et al. address such a focus more directly. The latter emphasize that computer professionals have a responsibility in anticipating the impact, and possible consequences of failure, of introducing computer-based artifacts, and that students of computing therefore should learn to appreciate the philosophical views and aesthetic values that play an important part in the development of the discipline. Anderson et al. focus directly on both moral and political issues.

Professional Approach

North American employers and business executives, says Denning, are dissatisfied because computing graduates lack practical competence. They cannot build useful systems, formulate or defend a proposal, write memos, draft a simple project budget, prepare an agenda for a meeting, work in teams, or bounce back from adversity; they lack a passion for learning. The current concept-oriented curriculum is well suited for preparing research engineers, but not the practice-oriented engineer on which competition increasingly depends [3].

The environment of computing has changed, and we need to change our perception of what it means to work as a computer professional. Denning suggests the conventional understanding of professional work as a set of tasks by which a group of people carry out a mission has become inadequate. Instead, we should perceive work as a closed-loop process by which a performer completes action leading to the satisfaction of a request by a customer or client. Computer professionals are not performing tasks to meet abstract objectives. They work for, or with, other people striving to meet their changing needs and requirements, and many of the skills the students lack are in the areas of communication and collaboration, rather than in technologies [3].

In the same vein, Turner et al. identify three fundamental processes: theory, abstraction, and design. The new curriculum attempts to integrate these fundamental processes and the social context of computing [10]. The conventional emphasis on theory is complemented with new perspectives on the importance of laboratories in the curriculum to strengthen abstraction and experimentation. The curriculum also develops communication skills and includes significant design experiences, such as working in teams.

Hartmanis et al.—being primarily concerned with the overall policies of computing—are rather vague when it comes to the actual practices of computer professionals. One of their important points, however, is that the strong connections between research and computing practices imply that the traditional separation of basic research, applied research, and development is somewhat dubious [4]. They recommend increased interaction between academia, industry, and society at large to enhance the cross-fertilization of ideas within computing between theoretical underpinnings and experimental experience [4].

The ethics code provides abstract and general rules for professional conduct; it does not prescribe guidelines. It does, however, imply a certain repertoire of skills and techniques that a computer professional must master. To give one example, computer professionals should give comprehensive and thorough evaluations of computer systems and their impacts, including analyses of possible risks. As a consequence, they are in a position of special trust, and have a special responsibility to provide objective, credible evaluations to employers, clients, users, and the public [1].

In terms of the three professional approaches of computing (see the sidebar "Professional Approaches") the targeted articles express positions

Professional Approaches

Construction approach: The task is to develop a technical artifact in response to a given and well-defined problem. To cope effectively with the complexity of this task, a rational approach is taken in which a sequence of specifications is transformed from overall requirements, through various levels of design, to a final, optimal solution.

Evolution approach: The task is to develop a technical artifact for a client or user with more or less clear and stable requirements. To cope effectively with the uncertainty of this task, an experimental approach is taken in which various models, prototypes, and versions are tried to reach a satisfactory solution.

Intervention approach: The task is to change a problematic situation in an organization through design and implementation of a computer-based artifact. The major challenge is to transcend traditions while at the same time protecting the good qualities of established work practices. This is done through an iterative learning process in which various solutions are designed, tested, and negotiated [2, 8].

in which an evolutionary approach is placed on equal footing with a more conventional construction approach. In fact, evolution is seen as a key approach to computing: practice is a closed-loop process by which a computer professional completes actions leading to the satisfaction of a request by a customer or client [3]; the fundamental processes of professional education are abstractions based on experiments and design, in addition to theory, and to facilitate these processes we should use laboratories in the curricula [10]; we need to increase interaction between academia, industry, and society at large, and to enhance the cross-fertilization of ideas within computing between theoretical underpinnings and experimental experience [4].

The intervention approach seems to play no role—or at least only a minor role. The implication of much of the rhetoric is, however, that an intervention perspective is needed to comprehend modern computing needs and practices. It takes the appreciation and skill of an intervention approach to develop computer-based artifacts that enhance the quality of working life, to thoroughly evaluate possible impacts and risks, and to ensure that the requirements and needs of different interest groups are taken into account and eventually met [1]. Similarly, Denning's program includes many aspects of organizational intervention. In addition to being competent in engineering, the computer professional should be a skilled listener for concerns of customers or clients, be rigorous in managing commitments and achieving customer satisfaction, and be organized for ongoing learning [3].

Professional Role

Turner et al. provide new perspectives on a number of key issues, one being the importance of social, ethical, and professional issues in computing curricula. They identify a body of subject matter representing the social and professional context of the discipline that is considered essential for every undergraduate program. Students of computing should develop the ability to ask serious questions about the social impact of computing and to evaluate proposed answers to those questions, and they must be able to anticipate the impact of introducing a given product into a given environment. To do so, they need additional experiences that will help them develop the capacity for critical thinking, problem solving, research methods, and professional development [10].

Similarly, Denning suggests that we should recognize a second kind of knowledge besides procedural knowledge. This second kind of knowledge includes knowing how to listen, to design, to persuade, to be organized for new learning, to be professional, and even to be trustworthy and honest. Both kinds of knowledge are essential for an engineer. Moreover, in emphasizing the important role of innovation, Denning sees an innovator as a person or organization that articulates a change, offers the means to do it, and mobilizes people to adopt the new practices. Innovation is an organizational phenomenon, not merely an individual one [3].

Hartmanis et al. are rather vague when it comes to specific views on the roles of computer professionals, but one can infer underlying changes in direction of a broader, more interdisciplinary professional profile. The code of professional conduct is, in contrast, quite explicit in its perspective on this aspect of our professional identity. The specific imperatives cited earlier all express the view of a socially concerned and responsible professional who actively attempts to enhance the quality of working life. This view is further elaborated in the general moral imperatives.

Computer professionals should contribute to society when designing or implementing systems. Computer professionals must attempt to ensure that the products of their efforts will be used in socially responsible ways, will meet social needs, and will avoid harmful effects on health and welfare. They also have a duty to be honest and trustworthy about their qualifications, and about any circumstances that might lead to conflicts of interest [1].

In our framework there are three different roles that computer professionals can play. None of these roles need to be less moral than the others, or less socially concerned. The difference between them lies rather in what is considered the most important factor to attend to if we want to improve the world: wealth, understanding, or equality.

The writings in *Communications* express positions in which the traditional role of the engineer is being supplemented with other values and responsibilities, most often emphasizing the role of facilitator but to some extent also the role of emancipator.

All texts clearly identify the facilitator role, and emphasize its importance and relevance to the computing profession. Computer professionals should, according to Denning, not only see themselves as traditional engineers who introduce new computer artifacts to make a set of actions more effective, but, in addition, as innovators working in teams to help organizations change their standard practices. Anderson et al. formulate the ideal of a computer professional who helps users express their needs and ensures that requirements are clearly articulated and implemented [1].

The role of emancipator is only vaguely expressed in the writings, except in the code of professional conduct. Here we find the image of a computer professional who is socially concerned, aware of the conflicts of interest related to the use of computers, strives to improve the quality of working life, and who generally contributes to society and human well-being. The computing professional who lives up to the new code of ethics and conduct is actively protecting and emancipating people from oppressive use of computing technologies.

Professional Roles

Engineer role: Engineers try to improve the world by developing better computer artifacts. They are mainly interested in technical knowledge that gives them superior control over the processes of computing. Engineers want to increase the efficiency of computing and computer use.

Facilitator role: Facilitators try to increase the competence of users and clients, handing over to them the responsibility of acting. They are mainly interested in contributing to a better understanding of the use of computing technology. Facilitators strive to increase our understanding of how technology could be made to serve people rather than the other way around.

Emancipator role: Emancipators try to use computing technology as an opportunity to advance society and social organizations. They are mainly interested in protecting and emancipating people from oppressive use of computing technologies. They worry about the role of computing technology in supporting injustice and unequal distribution of power [2, 5].

Computers in Use

We can summarize the message of the recent discussion in the ACM thus: in addition to the traditional technical competence of an engineer, a computing professional today needs more social, organizational, and communication skills. A vision of dual competence is presented, in which each engineer is both engineer and manager, salesperson and organizational expert, an ethically, socially responsible engineer, and so on.

This is fine. We have no quarrels with these suggestions. Broadening professional competence is necessary, and it is already a fact in the sense that engineers have performed as managers, in sales and marketing, working with organizational change, facing social responsibilities, and so on, even if they often have not been prepared by their education for these tasks. One can debate what is the best way to prepare a student for these tasks. Given a certain time frame, should students concentrate on core engineering skills, making sure that they become confident in their professional identities, or should students spend more time preparing for their actual tasks? This discussion echoes a more general dilemma confronted by all education in a complex society, and often the solution is compromise and we will have to continue debating, and changing, the proportions in that compromise.

We don't think, however, that the suggestions in the considered articles are sufficient. Turner et al. and Hartmanis et al. remain purely within mainstream thinking when it comes to their specific recommendations; Denning offers a critical analysis, but his recommendations are mainly supplements to, or modifications of, established traditions (broaden our research agenda, reformulate our curriculum around exhibitions of competent performance, reformulate our means and measures of assessment, provide institutional support for faculty development, and provide a more modern information infrastructure [3]. The new code of professional conduct implies a more radical view of our profession, but the imperatives are (naturally) so general as to give very little information on how to live up to them.

We must ask ourselves whether the proposed recommendations are effective responses to the requirements that seem to be imposed on our profession. What changes are needed and what should we as a profession do to facilitate these changes? Simply put, we argue that rather than envisioning a new engineer with social skills in addition to technical skills, we have to change what we consider technical competence.

In traditional engineering we concentrate on how artifacts function, and on how to make them function. The new engineering, we envision, will take a different view, attending to the use of artifacts, to the roles they play in our lives and how they play those roles. Such a perspective will revolutionize engineering, embedding it in a social context, making artifacts *in use*, rather than artifacts, its subject matter.

Sometimes, attempts at introducing more social and humanistic themes into the engineering curriculum seem to be an expression of an ambition to reduce the power of technology. This is not our vision. Technology is an outstanding social force and no amount of humanism will change that. But just because it is such a force, is it important for us, who have the ability to control it, to take an interest in its use.

A quick illustration of what we mean is the impact that a higher level of environmental awareness is beginning to have on engineering professions working with technologies, which have direct effects on the physical environment and on people as physical organisms.

Initially, the typical engineering response was to think of environmental issues as consequences of technology, leaving those consequences to politicians and other decision makers to deal with. As public interest in environmental issues grew, those issues began to interfere with engineering work, causing irritation. Gradually, however, environmental issues have entered engineering curricula, influencing the nature of technical competence itself in more and more branches of engineering. From being a source of irritation, pollution has become an area of engineering expertise.

Analogously, we argue that as technology becomes more and more important and pervasive in all aspects of modern life, engineers will have to include more and more social aspects in their technical problem solving and into the very core of the technical curriculum. It would be much easier and much less dangerous to introduce such aspects as additional, but distinct, skills, of course; but it would also be much more inefficient and contrived to attempt to do so.

Table 2. Key concepts of an ACM Computing Curricula [10]. The numbers in parentheses indicate the proposed lecture hours per subject area.

Fundamental Processes	theory; abstraction; design
Recurring Concepts	binding; complexity of large problems; conceptual and formal models; consistency and completeness; efficiency; evolution; levels of abstraction; ordering in space; ordering in time; reuse; security; trade-offs and consequences
Subject Areas	algorithms and data structures (47); architecture (59); artificial intelligence and robotics (9); database and information retrieval (9); human-computer communication (8); numerical and symbolic computation (7); operating systems (31); programming languages (46); software methodology and engineering (44); social, ethical and professional issues (11)

Our main argument is simple. When technology played a less pervasive role in society, as long as engineers were engaged primarily in military affairs, or in heavy industry; as long as they were not engaged in the everyday affairs of everyone, it was possible to carry on as if technology was somehow different from society. It made sense to speak of the social consequences of technology, and engineers became experts designing, constructing, maintaining, and repairing, technology while knowing next to nothing about the actual details of its use. But all this is changing now that technology is interwoven into everyday life.

The efficiency and productivity of modern society is based on the division of labor. Without division of labor, there is no hope of expertise. It has often been a long and arduous task—the different branches of engineering are good examples—to define and purify the particular areas of expertise. But in a changing society, lines of division will have to be redrawn, and even such "natural" dividing lines as those defining technical competence may have to be adjusted.

Computer technology, or as it is nowadays often called, information technology, is a particularly striking example of the need for change. The name "information technology" makes some of us a bit nervous. As computer professionals we work with computer technology. It is the users, politicians, and media that talk of information technology. How do the two relate to one another? When we develop computer technology, do we also develop information technology?

Really, that question puts the finger on the transition we are advocating. As long as our professional task was restricted to the machine itself, we were doing fine as traditional engineers. As long as computers were used as automata, it was their independence from human beings that made them so powerful. The power of information technology, on the contrary, lies in its dependence on human beings, in the many ways—as tools, networks, media, and the like—in which it involves and enhances human actions and interactions. It is this power of information technology to infiltrate our lives and our minds that places new demands on our profession.

Consequences for Computing Curricula

These are big issues, and in order to result in something more than hot air, visions will have to be turned into specific programs for action. We can begin by taking a look at our curricula from this perspective and compare what we would do to the suggestions given by Turner et al.

Turner et al. propose to change the computing curriculum. As summarized in Table 2, they identify, using their own terminology, three processes and twelve recurring concepts that should be fundamental to the computing discipline as a whole, and they identify ten subject areas that should comprise the subject matter of the discipline. The subject areas listed in Table 2 appear to be the result of a bargaining process between established traditions. As a consequence, it is terribly conservative and skewed (half of the subject areas—algorithms and data structures, architecture, operating systems, programming languages, and software methodology and engineering—cover 84% of the lecture hours). Nontraditional engineering subjects are added as isolated subjects (e.g., "social, ethical, and professional issues") and little weight is assigned to such subjects (4% of the lecture hours).

A number of questions can be raised regarding the proposed subject areas: Why is programming not treated as a subject area or a fundamental process in its own right (instead, programming occurs in all of the first nine subject areas)? Why are operating systems, as a subject area in its own right, and compilers, as an important part of the programming language subject area, still the only types of applications of computing that are given in-depth treatment? Why does the subject area "programming languages" not include specification languages in general as well as development environments? Why not include subject areas like programming and modeling, languages and environments, application paradigms, and computers as media?

Two of the recurring concepts of computing, proposed by Turner et al., go beyond a traditional approach to computing. One is the concept of "evolution," needed to deal with evolutionary changes and their implications for all levels of computing. Another is "trade-offs and consequences," to be used in handling trade-offs in computing, the consequences of such trade-offs, and the technical, economic, cultural and other effects of selecting one design alternative rather than another [10]. The traditional orientation of the recurring concepts is, however, illustrated by questions like: Why "complexity," but not uncertainty or risk? Why "efficiency," but not effectiveness? Why is quality not a recurring concept? Why are important dichotomies like data and information, and process and structure not included?

The three fundamental processes proposed by Turner et al. are characterized in the following way: *theory* is a process rooted in mathematics and used to develop coherent formal theories; *abstraction* is a process rooted in the experimental sciences and used to develop conceptual understanding; and *design* is a process that is rooted in engineering and used to develop computer-based artifacts to solve given problems [10]. These concepts raise a number of fundamental questions: Why is "theory" reserved for formal theories rooted in mathematics? Why is "abstraction" considered to be distinct from "theory"? Why are experiments only related to "abstraction"? Why is "design" only rooted in engineering? Why is "design" restricted to solving given problems? Based on such questions, a different conception of the fundamental processes of computing emerges:

Theory: This process is rooted in scientific disciplines, such as mathematics and organizational behavior, that are fundamental to computing. We use this process to develop theories and conceptual frameworks to understand, design, and evaluate computer-based artifacts in use.

Design: This process is rooted in design disciplines, such as architecture and industrial design, that share with computing an interest in artifacts in use. We use this process to develop specific design skills and the ability to organize and manage experiments.

Interpretation: This process is rooted in the humanities, in anthropology, and history. We use this process to understand and evaluate artifacts in use and problematic situations in computing practices.

Table 3. Key concepts of an alternative curriculum

Fundamental Processes	theory; design; interpretation
Recurring Concepts	data and information; quality; complexity and uncertainty; conceptual and formal models; consistency and completeness; efficiency and effectiveness; binding; evolution; levels of abstraction; ordering in space; ordering in time; reuse
Subject Areas	computing machines, architectures and networks; information systems, database management and information retrieval; control systems and operating systems; personal computing, human-computer interaction and interface design; artificial intelligence and robotics; programming, algorithms and data structures; software methodology and engineering; programming languages and computer linguistics; multimedia, intelligent agents, and Internet technologies.

In this discussion of the subject areas, recurring concepts, and fundamental processes of the "Computing Curricula 1991" we have begun to formulate a new program for the education of computer professionals, beginning with the use of computers, with an interest to improve both technology and its use. In more specific terms, such a program would:

Focus on use: Give the students a chance to develop a conscious and critical attitude as users of computer systems; teach them how to evaluate and compare the use of different types of systems; experiment with various forms of collaboration between users and computer professionals; give them a deep understanding of the differences and similarities between major types of computer applications.

Emphasize interpretation: Introduce the students to a spectrum of (qualitative and quantitative, informal and formal) approaches to describe, evaluate, and communicate about the design and use of computer systems; make them conscious of the role of the observer and the perspectives underlying different approaches.

Integrate perspectives: Encourage the students to use and integrate different perspectives, stressing the importance of being able to view the technology from the user's point of view; address natural and formal languages together to give a general understanding of languages as related to understanding, design and use of computer systems and to help students appreciate the relative strengths and weaknesses of different forms of expression and communication; distinguish between support and control, drawing attention to the role of computer applications as instruments of control.

Change priorities: Focus on use, and on dichotomies like information and data, process and structure, complexity and uncertainty, efficiency and effectiveness, rather than on a traditional, one-dimensional framework based on data, algorithms, architecture, complexity, and efficiency.

Emphasize quality: Make the notion of quality an integrating, recurring concept, encouraging the students to develop a critical, but constructive attitude to the design and use of computer systems, and inviting them to reflect on their profession from a practical, more holistic point of view across traditional subject areas and academic disciplines.

Using the schema introduced by Turner et al., these considerations result in a curriculum such as the one outlined in Table 3, where the subject areas to a large extent are defined in terms of the various ways in which computers can be used.

On a more general level, there is—as pointed out by Denning [3]—a growing doubt about the effectiveness of educational systems, and as companies loosen their bureaucratic ties in favor of networking, people are beginning to speak about "the virtual university." In Scandinavia, we are now experimenting with extensive programs for "life-long learning" and "student projects," introducing a more practically oriented curriculum running parallel with the theoretical one, "professional networks," involving professionals and professors in schemes of job rotation and joint seminars, and "collaborative, virtual research centers," bringing universities and software organizations together in joint projects. Nothing about this is new, of course, as pointed out recently by Norman et al. in their discussion of learner-centered education [9]. But the attitude is different. The transcendence of traditional boundaries between education and work indicate that the industrial-age educational system, with the school as just another type of factory, is ready for revolution, after all.

Conclusion

The computing profession has grown out of mathematics and engineering departments, and we continue to seek our identity in these departments. We see ourselves as experts on the mechanism itself, on computer architectures, operating systems, compilers, programming languages, and database management systems. Over the years, the attention of our profession has shifted from numerical analysis to programming to software engineering to human-computer interaction to networking. These shifts have added new elements to our professional competence, slightly changed the center of gravity of our profession, but otherwise left many of us undisturbed.

The change that we have advocated here is more radical. It means introducing a romantic use perspective into every aspect of teaching and practicing our profession. Mechanistic thinking, so powerful in producing and characterizing the machine, may actually hamper us when we are trying to put it to good use. Even if acquiring a more romantic perspective may make no immediate difference to one's work habits as a programmer in the software industry, it is important for one's general role as a computer professional. Whoever is working with a technology with a great impact on people risks having to address questions concerning that impact.

We have illustrated the effects of the change we are advocating by outlining an alternative to the "Computing Curricula 1991" [10], and by suggesting a change of the educational system as such, exemplified by some experiments currently under way in Scandinavia and designated by concepts like learner-centered education.

Rob Kling has proposed an approach to educating computing professionals in which organizational informatics plays an important role [6, 7]. Kling argues that students should acquire reliable knowledge about the social dimensions of systems development and use and such educational efforts should build upon both the traditional technological foundations of computer science and the social sciences. Kling's proposal goes further in responding to the emerging requirements on our profession, but it shares with the "Computing Curricula 1991" the idea of adding new features to existing disciplines. We have indicated a different option: to redefine our notion of technical competence based on an interest in the use of technology—not with the purpose of rejecting useful technical knowledge, but with the ambition to challenge and eventually change the very basis of our profession. C

Acknowledgments
We are grateful to Sten Henriksson and Finn Verner Jensen for valuable and useful comments on previous versions of this article.

References

1. Anderson, R.E., Johnson, D.G., Gotterbarn, D. and Perrolle, J. Using the ACM Code of Ethics in decision making. *Commun. ACM 36*, 2 (Feb. 1993), 98–107.
2. Dahlbom, B. and Mathiassen, L. *Computers in Context: The Philosophy and Practice of Systems Design*. Blackwell, Cambridge, Mass., 1993.
3. Denning, P.J. Educating a new engineer. *Commun. ACM 35*, 12 (Dec. 1992), 83–97.
4. Hartmanis, J., et al. Computing the future. *Commun. ACM 35*, 11 (Nov. 1992), 30–40.
5. Hirschheim, R. and Klein, H.K. Four paradigms of information systems development. *Commun. ACM 32*, 10 (Oct. 1989), 1199–1216.
6. Kling, R. Broadening computer science. *Commun. ACM 36*, 2 (Feb. 1993), 15–17.
7. Kling, R. Organizational analysis in computer science. *The Information Society* 9, 2 (Mar.–June 1993), 71–87.
8. Lanzara, G.F. The design process: Frames, metaphors and games. In Briefs, U., Ciborra, C. and Schneider, L., Eds. *Systems Design For, With and By the Users*. North Holland, Amsterdam, 1983.
9. Norman, D.A. and Spohrer, J.C. Learner-centered education. *Commun. ACM 39*, 4 (Apr. 1996), 24–27.
10. Turner, A.J., et al. Computing curricula 1991. *Commun. ACM 34*, 6 (June 1991), 69–84.

Bo Dahlbom (dahlbom@adb.gu.se) is a professor of Informatics at Göteborg University, University of Aalborg, Aalborg, Denmark.
Lars Mathiassen (larsm@iesd.auc.dk) is a professor of Electronic Systems at the University of Aalborg, Denmark.