Out of Scandinavia:
Alternative Approaches to Software
Design and System Development

Christiane Floyd, Wolf-Michael Mehl,
Fanny-Michaela Reisin, Gerhard Schmidt, and
Gregor Wolf
Technical University of Berlin

ABSTRACT

This study set out to delineate the Scandinavian Approach to the development of computer-based systems. We aimed to help derive new ideas for human-oriented technology design in other countries. The study is based on the relevant literature, scientific contacts, and two field trips, and covers work in Denmark, Norway, and Sweden.

The study focuses on methodological questions and their theoretical foundations, on explicit strategies for social implementation, and on innovative design illustrated by reference to concrete projects. Though it makes no claim to present a sociopolitical analysis of Scandinavian technology design, the sociocultural background is given due consideration.

There is no general agreement among Scandinavians as to whether or not there is a well-defined Scandinavian Approach. We have come to identify such an approach in certain common features shared by the different schools of thought. These include efforts toward humanization and democratization as overriding design goals, in keeping with the aim of building an egalitarian society.

This article is descended from a report in German. It largely follows the original German text which appeared in early 1987 and was based on research done in the latter half of 1986.

CONTENTS

1. INTRODUCTION
2. TECHNOLOGY DESIGN IN A SCANDINAVIAN CULTURAL SETTING
3. FOUNDATIONS FOR THE DESIGN OF COMPUTER-BASED SYSTEMS: AN OVERVIEW
   - The Spirit of the Scandinavian Approach
   - Scandinavian Traditions in Applied Computer Science
   - Development of Ideas on Work Design
   - Historical Shifts
   - The Systems Perspective—and Beyond . . .
   - Software Development as Design
   - Controversies and Agreement in Scientific Practice
4. ORGANIZATION-ORIENTED AND WORK-ORIENTED APPROACHES
   4.1. The Sociotechnical Systems Approach
   4.2. The Collective Resource Approach to Computer-Based Work Design
         NJMF—Norsk Jern og MetallarbeiderForbund
         DEMOS
         DUE—Demokrati, Udvikling og EDB
         “Second Generation” Projects
         Basic Assumptions Behind the Collective Resource Approach to Systems Design in Scandinavia
5. SYSTEM-ORIENTED APPROACHES
   5.1. Development of Information Systems: The ISAC Methodology
         Description
         Evaluation
   5.2. Object-Oriented System Description
6. SOFTWARE DEVELOPMENT AND ITS METHODOLOGICAL SUPPORT
   6.1. Datalogy: Programming Viewed as Theory Building
   6.2. Methodological Work Forms in Systems Development
         Dialectical Approach
         Theory of Activities in Systems Development
         Methods and Techniques
   6.3. Cooperation Between Developers and Users
         Mutual Learning in Participative System Design
         Prototyping—A Methodical Approach to Evolutionary System Design
7. TOWARD A THEORY OF SYSTEMS DEVELOPMENT
   The Nature and Scope of Informatics
   Perspectivity in Scientific and Professional Practice
   The Adoption of Perspectives in the Context of a Development Project
   The Significance of Perspectives in Connection With Methods for Software Development
   Consideration of the Domain of Informatics From the Perspective of Different Scientific Disciplines
   The Value-Guided Adoption of Perspectives on Technology Development
   With a View to Attaining Specific Sociopolitical Goals
   The Emergence of the Research Program “Computer Support for Cooperative Work”
8. DEVELOPMENT PROJECTS—CASE STUDIES ILLUSTRATING NEW APPROACHES

8.1. Municipal Town and Country Planning—Enhancing User Skills With the Help of Prototyping
   Key Questions
   System Development Strategy
   The Development Strategy
   Results

8.2. Florence—Supporting Communicative and Cooperative Work Processes
   Key Questions
   Methodological Approach
   The Development Process
   Results

8.3. UTOPIA—Design of Computer-Based Systems for Enhancing Work Quality and User Skills
   Design Methods and Techniques
   Methodological Support
   The Project Itself

9. CONCLUSIONS

APPENDIX: POTENTIAL CONTACTS IN SCANDINAVIA

The theoretical foundations for the development of computer-based systems draw on computer science, organization theory, and the humanities. Specifically Scandinavian traditions in computer science provide the technical basis for discussions on work design. The most important schools in Scandinavia are the sociotechnical and the union-oriented collective resource approach, each associated with a distinct strategy for societal implementation.

The Scandinavian Approach can be seen as a challenge to other countries, combining as it does technical sophistication with the explicit concern for using computers for the benefit of human beings.

1. INTRODUCTION

A study of the distinctive features of computer-based system development and use in Scandinavia could well turn out to be a life work or at least run to several volumes. In contrast, the pretensions of this study are comparatively modest. We adopted a selective approach, proceeding from our own particular viewpoint, making use of already existing contacts or establishing new ones, insofar as this was possible in the time available. Our chief objective in doing so was to study alternative trends in software development in Scandinavia with a view toward deriving new ideas for human-oriented technology design in other countries. In pursuing this objective, we were immediately confronted by the following questions:
• Is there such a thing as an independent Scandinavian Approach to computer-based system design in Scandinavia, and what are its special features?

• If so, to what extent can it be divorced from the context of Scandinavian culture and be fruitfully applied in another context?

To make the study more easily accessible to the reader, we include in this introduction details of our own personal background, of the approach adopted in preparing the study, and our own views on the aforementioned questions. We then explain the structure of the subsequent sections.

We are all members of the research group on Software Engineering, headed by Christiane Floyd, at the Computer Science Department of the Technical University of Berlin. Since 1978, we have been working on a methodological approach known as Software Technology for Evolutionary Participative System Development (STEPS; Floyd, Reisin, & Schmidt, 1989), which is primarily concerned with the development of software as an adequate tool for users. Elaborating this approach has involved and continues to involve fundamental research, necessarily transcending the conceptions of software engineering prevalent in the Federal Republic of Germany. These see it as being based on fixed requirements of the software products to be developed, ignoring the social context in which they are embedded.

The dialogue with scientists and researchers in Denmark, Norway, and Sweden—with whom we have been in constant touch since 1982—has proved most fruitful. Several international working conferences have also provided a suitable forum for an intensive exchange of ideas.1 This explains the substantial degree of concurrence between our basic views and those embodied in the work of Scandinavian scientists.

This fruitful exchange of ideas on a professional level has been underscored by a long-standing and thoroughly positive relationship to Scandinavia. Existing professional contacts must, then, be seen in light of personal ties of friendship with several of the authors cited in this study and with high regard for Scandinavian culture. Our aim to is help promote a more intensive interaction based on a professional exchange of ideas and personal contacts with a view to encouraging, on all sides, a technological development taking account of human needs.

Our choice of material for this study must be viewed with reference to both

---

1 For example, the International Working Conference on Prototyping, Namur, 1983 (see Budde, Kuhlenkamp, Mathiassen, & Züllig, 1984); the International Conference on the Development and Use of Computer-Based Systems, Århus, 1985 (see Bjerknes, Ehn, & Kyng, 1987); and the International Conference on System Design for Human Development and Productivity: Participation and Beyond, Berlin, GDR, 1986 (see Docherty, Fuchs-Kittowski, Kolm, & Mathiassen, 1987; Fuchs-Kittowski & Gertenbach, 1987).
the specific topic under consideration here and the overall situation in which the study was prepared. The study is not a sociopolitical analysis of Scandinavian technology design. The sociocultural background is outlined and given due appreciation, but it is not the central topic of interest here. Furthermore, there are many large-scale projects being conducted in important industrial enterprises and other organizations that cannot be covered here, despite their substantial implications for the practical application of information technology in Scandinavia and the fact that a study of their findings might prove highly instructive. In contrast, our study focuses on methodological aspects, considering questions along the following lines:

- Which theoretical foundations and practical methods are employed in Scandinavia?
- How are these embedded in the sociocultural background?
- How are they effective in pursuing value-guided technological development?

The citations from the work of particular authors included in our study, together with descriptions of selected projects, serve to illustrate and elucidate these questions. We have chosen several particularly relevant methods, but we do not claim to have made an exhaustive survey of methods. We singled out three case studies in order to illustrate specific concerns, without wishing to suggest that these cover the most important, let alone the only important, projects being conducted.

Scandinavia is taken to mean Denmark, Norway, and Sweden. There are, of course, many significant differences among these three countries, due to historical and geographical factors, as well as to their respective cultural backgrounds and their exchanges with other cultures. We attempt to avoid superficial generalizations, but it is impossible to delineate these differences in detail, let alone try to substantiate them. Looking at the differing political situations in these three countries in the years following World War II (a period of crucial importance for the emergence of computer technology), the differing legislative traditions, or different institutional approaches to sociopolitical issues, we are compelled to pass over a number of important questions. Any endeavor to put the subject of our study into a historical perspective must be a tentative one.

The information on which we based our study was obtained by intensifying existing contacts, by examining the relevant literature, and by undertaking two field trips to Scandinavia. Distinct differences became apparent between Denmark and Norway on the one hand, both small countries with which close contacts had already been established, and Sweden on the other, whose size alone made it impossible to cover all aspects of interest. But then, our prime
concern was not to produce an exhaustive survey, but rather to draw a
detailed picture of relevant methodological approaches within the chosen
area, approaches whose influence was either confined to a specific country or
transcended national borders. To this extent, we hope that Sweden, too, has
been given sufficient consideration in this study.

In the numerous discussions held during our field trips, our first question
was invariably: Is there such a thing as a Scandinavian Approach to
computer-based system development? We received a wide variety of answers
to this question. One of the schools (which has grown up around Kristen
Nygaard) is sometimes called the Scandinavian Approach, yet other profes-
sionals we spoke to were of the opinion that there is no well-defined
Scandinavian Approach. Many of them claimed to be pursuing quite
ordinary technology development, which is no doubt the case. A third view
was that the specific characteristics of the Scandinavian Approach are to be
found in the values shared by the system developers or in the common goals
of system development.

We came to the conclusion that there is a Scandinavian Approach, but that
it is not necessarily a tangible phenomenon for the Scandinavians themselves.
The reason for this is that, in our opinion, it consists in certain common
features of the different schools, which we consider to be intrinsic, but which
are taken for granted in their own cultural setting to the extent that they are
not consciously recognized as such. Conversely, the Scandinavians are able to
put their finger on the differences between the respective approaches, and in
some cases these are the subjects of controversy and discussion.

In our view, the Scandinavian Approach is not merely a question of values
shared by system developers; it is also connected with several other sociocul-
tural factors, which are examined in Section 2 of the study.

Section 3 looks at methodological endeavors from the field of computer
science\(^2\) as well as those pertaining to organization and work design. These
are often related to one another. In some cases they complement and are in
harmony with each other, whereas other efforts are in competition.

Sections 4 to 7 present individual approaches and their development in
more detail.

Section 8 looks at three different attempts to translate the concern with
socially oriented design into concrete terms in practical project situations
using innovative methodological approaches.

In Section 9, the concluding section, we focus on certain factors that we
consider relevant if we attempt to draw conclusions from the Scandinavian
approaches and apply them to other cultural settings.

\(^2\) A number of different terms are or have been in use in Scandinavia to designate this specific
field. To simplify matters, we have used the term “computer science” throughout, except in cases
where a particular author favors a different term.
2. TECHNOLOGY DESIGN IN A SCANDINAVIAN CULTURAL SETTING

In order to do justice to the distinctive features of the Scandinavian approaches to socially oriented design of computer-based systems, it is essential to give at least a brief outline of the way in which these approaches are embedded in the sociocultural traditions of the Scandinavian countries. Crucial factors here are, first, the way in which the respective societies see themselves today and their respective historical situations during the period when information technology was emerging, and, second, the form cooperation takes in these countries, the prevalent conception of science and its practical application, and the attitude to technology as a whole.

When attempting to elucidate for outsiders the specific characteristics of these approaches, Scandinavians themselves frequently delve far back into their own history. In the interviews for this study, our attention was drawn twice to the fact that feudalism never existed in Norway and Sweden (with the exception of the former Danish possessions); feudalism did, however, exist in Denmark, but it was abolished at the nobility's own instigation.

Thus, factors deeply rooted in the history of this region would have to be systematically followed up to the present time in order to show (a) how the background against which computer technology emerged in Scandinavia was, from the outset, different from that in other countries and (b) how distinct social mechanisms were at work, influencing the practical implementation of this technology.

We do not consider it our job to give a detailed outline of Scandinavian history, and we are not qualified to do so. We do feel, though, that various factors, both individually and in combination, have gone a long way toward shaping the developments to be described in the following sections, and their implementation in a social context. It is impossible for us to give a complete picture of a "cultural setting"; we can only attempt, by reference to our own cultural background, to focus on factors that are striking or different and appear to have a direct and obvious bearing on the subject of interest. The following brief historical outline is to be understood in this context.

An essential feature in Scandinavia is, above all, what appears to outsiders as a far-reaching and widely supported fundamental concern with the building and development of a society in which each individual may live in dignity and in conditions conducive to personal development. This means that, despite existing conflicts of interest, different social groups appear to be pledged to this common goal. A keen endeavor is perceptible to make optimal use of existing material resources, devoting great care to the question of design, in order to achieve a high degree of quality. The combined skills of all members of the society are considered necessary to attain this overriding objective. Central European observers like ourselves are struck by the
fundamental will to establish a social consensus and to bring about a
deliberate activation of human resources with a view toward shaping a society
that offers a high quality of life to all its members and largely compensates
social injustices, that is, to create an egalitarian society.

It seems natural to view these efforts to establish an egalitarian society as
being closely connected with the specific conditions of life in the North in past
centuries. In a society consisting of a small, widely scattered population
seeking to eke out a living under extremely hostile natural conditions, there is
a strong sense of being dependent on one another. Under such circumstances,
hierarchical power structures with glaring class distinctions were unable to
gain as firm a hold here as elsewhere (take, e.g., the absence of feudalism).
Widespread poverty and hardship appear here, to a greater extent than
elsewhere, to be due to the inclemencies of nature that have to be withstood
together as a community, rather than to social oppression and exploitation,
which lead to violent confrontations between different social groups. It is
important that these assessments be seen as being of a relative, not absolute,
nature; it would be a mistake to try to establish an exclusive causality between
the two phenomena. In any case, what we have just said only really applies to
Norway and Sweden, though in Denmark, too, the same basic attitude to the
shaping of society can be encountered.

Another important historical factor can be seen in the armed conflicts
afflicting this region in the past over a period of several centuries. Although
these, on the one hand, drained the countries' resources and increased the
need for rebuilding their economies, they led, in the 19th century, to
territorial diminution in the case of Denmark and to Sweden's loss of military
influence. And they caused both countries, together with Norway, which was
at this time just emerging as an independent nation, to turn consciously
toward a policy of peace. In all three countries this encouraged the building
of an egalitarian society precisely during the period of industrialization. This
self-chosen course of peaceful development was brutally interrupted when
Denmark, and more particularly Norway, became embroiled in the tragic
events of World War II. It may be assumed that these bitter experiences have,
in the period following the wartime confusion, led all the Scandinavian
countries to redirect and increase their endeavors toward building (or
rebuilding) a peaceful, egalitarian society.

Efforts to cope with industrialization on a social level have, in keeping with
the fundamental endeavors just mentioned, also helped create a more
favorable climate for the serious consideration of workers' interests than
elsewhere. An Australian study of the situation in Scandinavia (Gunzburg,
1974) stressed the importance of the following factors:

- Scandinavian society is without a long industrial tradition.
- There exists a high capacity for negotiation and a will to solve
  problems collectively.
OUT OF SCANDINAVIA

- There are strong, centrally organized associations of employers on one side, and strong, highly organized trade unions on the other.
- Strong ties exist between the trade unions and the social democratic parties.
- Both the trade unions and the employers' associations demonstrate a high degree of cooperativeness.
- The attitude of the trade unions is an innovative and forward-looking one.

These have been decisive factors in bringing about legislation and agreements safeguarding workers' interests and according them far-reaching rights of codetermination. Progress in this area was to some extent inspired by international discussions, and it could serve as a model for other countries. Moreover, aspects such as job satisfaction, workplace design, and safety were brought into play at a very early stage and seen in their relation to economic demands. Ultimately, a number of research institutes\(^3\) in this area have grown up, some under government auspices, others supported by the trade unions or affiliated to universities; they have played a substantial part in developing alternative approaches to socially oriented technology design.

At the same time, the striving toward an egalitarian society, ensuring a high quality of life for all its members, has also called for the activation of all available human resources with a view to establishing qualified cooperation and bringing about a quality oriented improvement in social systems supported by technology. This is manifest in several specific developments that can be seen throughout Scandinavia:

- The particular importance of adult education following the emergence of adult education colleges in the 19th century (e.g., in Denmark, where they played a crucial role in agricultural reform). These institutions serve, among other things, to familiarize the working population with the possibilities for meaningfully utilizing modern technologies, and they enjoy a position of comparatively high prestige in Scandinavia.
- The emergence of design as a specific skill in handicraft work as a factor in various production sectors, but also as a feature encountered in areas of the social infrastructure like health or education. Design aims at reconciling different, sometimes conflicting concerns, making

---

\(^3\) In Norway, the Norwegian Institute for Social Research and the Norwegian Computing Center; in Sweden, the Swedish Work Environment Fund and the Swedish Center for Working Life.
optimal use of available resources in order to create high quality products or systems at a reasonable cost.

- The cooperation between different social groups in an effort to attain goals of overriding importance. Of particular interest for the purposes of this study are the following types of cooperation, which have reached a comparatively high level of development: interdisciplinary cooperation between scientists belonging to different fields (e.g., between computer science on the one side and organization theory and the humanities on the other), cooperation between scientists and industry on the basis of pilot projects, cooperation between scientists and trade unions, and the cooperation between trade unions and employers already discussed herein.

- The types of cooperation just described occur in a broad social context and have made a substantial contribution to developing and implementing the approaches under consideration here. The approaches themselves, however, are concerned with establishing a high degree of cooperation between groups involved in the narrower context of system development projects. Here, various forms of user participation in system design have been developed and tested.

- A fundamentally positive attitude to innovations in the fields of science and technology. These are seen as tools to help attain the objective of an egalitarian society offering a high quality of life. The antagonism between the advocates of ruthless technological progress in science and industry and their opponents who are attempting to defend social values in the face of the inexorable advance of technology is felt to be much less pronounced in Scandinavia than elsewhere. In fact, in Scandinavia attempts are made to overcome this antagonism so as to use technology to meaningfully support social systems, while safeguarding human and social values.

A final factor of importance with respect to the developments under consideration here is the opening up of the Scandinavian countries to Anglo-American influence. Particularly in Norway, this has a tradition going back several centuries, associated with this country's trade relations. Denmark, and to some extent Sweden as well, underwent a process of reorientation as a result of political events leading up to World War II.

This is of significance for the purposes of our study because, for one thing, the sociotechnical systems approach to work design found its way to Scandinavia via Norway almost immediately after its initial emergence in England and played a key role there as an inspiration for Scandinavian approaches to work design. Second, the importance of computer technology
as a whole for the scientific, industrial, and administrative sectors was recognized much earlier in Scandinavia than in many other European countries. This also entailed an early beginning of efforts relating to computer science.

To sum up, it may be said that the historical situation in Scandinavia in the early 1950s provided fertile soil for the growth of computer technology. During this period, Norway, in particular, but also Denmark were engaged in reconstruction programs; in these two countries, as well as in Sweden, full recognition was being given to the importance of innovative technologies as aids in building an egalitarian society capable of holding its own with respect to international competitiveness in the economic sphere while offering its citizens a high quality of life.

While Scandinavian scientists took part in the general international discussion, they also worked on distinct design-based approaches to using computers as tools for solving practical problems or as aids for supporting work processes in organizations.

The basic tendency toward an egalitarian society is reflected in the design of computer-based systems. Thus, in a comparison of such systems in Canada and Denmark (Bjørn-Andersen & Kumar, 1985), Nils Bjørn-Andersen, professor at the Copenhagen School of Economics and Business Administration and a leading advocate of the sociotechnical approach, underlined the fact that what the various Scandinavian schools have in common are the values shared by the system developers there. For them, humanization and democratization are overriding goals, which are given much greater attention in designing computer-based systems than elsewhere.

These goals have given rise to distinct scientific and methodological endeavors, drawing on theoretical foundations from organization theory and work design as well as from computer science, which are examined in the following sections. They are also associated with genuinely Scandinavian mechanisms of social implementation.

Controversies and criticism are quite common among the different schools of thought. In light of such controversies, we must recognize another important factor in Scandinavia affecting the social implementability of the approaches considered here: This is the fact that conflicts involving differing views and sometimes contrary convictions are voiced, argued out, and in some cases actually overcome, in an atmosphere of mutual respect and by maintaining an overriding will to cooperate.

The totality of these various interlaced endeavors, some of which directly interact, others of which are complementary or even in competition with each other, has resulted in a trend toward the design of computer-based systems taking careful account of the needs of human users. This phenomenon is what characterizes the Scandinavian Approach to outside observers.

Naturally, there is also plenty of ordinary technology being developed in
Scandinavia. The remarkable fact is, though, that the design-oriented efforts of interest here have been able to evolve and make themselves felt to such an extent in the society at large. Unlike the prevalent form of computer science, whose development has been strongly shaped by the United States and the military interests dominant there, the Scandinavian Approach has been largely shaped by nonmilitary needs and is explicitly geared to the use of computers for the benefit of human beings, in pursuance of the overriding goals of humanization and democratization.

Thus, the Scandinavian Approach may be seen as an alternative, or even as a challenge to other countries. And Scandinavian scientists of various schools who have been following the international discussion are becoming increasingly aware of this fact. It is no coincidence, then, that the proceedings of a major conference on the development and use of computer-based systems (held in 1985 at Århus University in commemoration of the 10th anniversary of the founding of the trade-union-oriented school that grew up there under the guidance of Kristen Nygaard, professor at the University of Oslo) were published under the title Computers and Democracy, a Scandinavian Challenge (Bjerknes, Ehn, & Kyng, 1987).

3. FOUNDATIONS FOR THE DESIGN OF COMPUTER-BASED SYSTEMS: AN OVERVIEW

In this study, we have the not-so-easy task of presenting a number of theoretical and methodological approaches that are either interlinked, or that have been developed alongside one another at various places in Denmark, Norway, and Sweden. These approaches have emerged over a period of approximately 25 years and have changed in step with advances in information technology, the varying socioeconomic climate alternating between growth and consolidation, and with increasing experience and insight. We set out to explore the interrelation between the applied computer and information sciences on the one hand and endeavors for human-oriented work design, supported by the social sciences, on the other.

We have chosen to write this section by way of an overview, because neither the choice of material nor the structure of the sections that follow are self-explanatory. The specific concern here is to show the interrelations between approaches and their development over time.

Subsequent sections are devoted to presenting individual approaches in more detail. They are structured primarily along the lines of the different approaches, rather than focusing on particular aspects of the scientific discussion. We consider this to be the best procedure, because our aim in most cases is to give an integral picture of existing approaches, such as may be derived from the available literature. This does, however, result in repetitions, overlaps, and cross-links between the different sections that may not be
**Figure 1.** Relative position of the Scandinavian Approach. *Note.* From an ad hoc sketch by Hans-Eric Nissen during our interview with him, October 1986.

Evident to the reader at first glance. We therefore feel the need to establish a historical and conceptual platform in order to make the subsequent sections more intelligible.

**The Spirit of the Scandinavian Approach**

The difference between the Scandinavian and other approaches to system development was pinpointed, in the course of one of our interviews, by Hans-Eric Nissen, professor at Lund University, using two pairs of alternatives to illustrate his point:

- The system should either reflect the interests of the system owners, or—as fairly as possible—the interests of all those affected (closely related to the goal of democratization).
- The system is primarily designed to compensate human weaknesses, or to support human strengths (closely related to the goal of humanization).

What the various Scandinavian schools have in common, and what distinguishes them from other traditions, is, according to Nissen, the fact that they tend toward the second of the two alternatives in each case (see Figure 1).
Furthermore, the distinctly Scandinavian attitude toward science and technology, which are seen as aids for designing social systems, has encouraged the development of a computer science tradition that is less technocentered and more closely geared to the application of computers in organizations. Thus, we are able to identify early traditions in computer science that provided the technical basis for work design.

**Scandinavian Traditions in Applied Computer Science**

In Sweden, the special field, information systems, has evolved from the pioneering work done by Börje Langefors, who began to look at the problems of systems development as early as 1943. Langefors’s theoretical and methodological considerations met with an overwhelming response in Sweden itself and, by the early 1960s, had found their way, via training courses, to Denmark and Norway, too, where they began to influence developments. The school of thought behind information systems was among the first to emphasize the importance of considering the information processes retained by human beings, as well as those to be automated. It concludes that the proper term to use here is computer-based information systems. The Information Systems Work and Analysis of Changes (ISAC) methodology looked at in Section 5.1 is a direct extension of Langefors’s work. And it was through the use of ISAC and other related approaches that, above all in Sweden, the internationally best known major projects on socially oriented technology design in the field of information technology were conducted.

In Denmark, computer science has evolved since the 1950s under the name Datalogy, and has been strongly influenced by the ideas of Peter Naur. Now a professor at Copenhagen University, Naur was at that time associated with Regnecentralen and engaged in pioneering work on programming languages (mainly ALGOL 60) and compiler construction. Beyond this, he has been instrumental in shaping the orientation of computer science and university education in this field in Denmark. Starting from his original concern with programming languages, he soon became engaged in programming methodology, focusing his attention, as early as the mid-1960s, on programming as systematic work and on programs as tools for people to solve practical problems. Since about 1970, he has formulated an empirically based critical appraisal of international efforts to establish a formalistically oriented programming methodology. In its central points, his criticism is concerned with how programmers are to proceed meaningfully in the software development process and provides far-reaching insights toward a holistic understanding of software development known as the theory building view of programming (see also Section 6.1). In Norway, the development of the programming language SIMULA in the 1960s must be considered the most spectacular achievement of computer science. It owes its origin to the collaboration between Ole Johan Dahl,
professor at Oslo University, and Kristen Nygaard, now also a professor at Oslo University, then at the Norwegian Computing Center. They developed powerful concepts for system description and simulation and combined them with standard programming constructs, using ALGOL 60 as a base language. Their work had a profound influence, not only on the ideas of computer scientists working in Scandinavia, but also on international developments in the field of programming languages. It provided innovative concepts for structuring parallel and sequential programs and for modelling conceptual hierarchies in classes. In retrospect, it may be considered the origin of object-oriented programming. It also inspired the development of concepts for system description for use in participative system development (see the review of DELTA and BETA in Section 5.2).

By contrast, the special field of software engineering, which emerged during the same period and addresses to some extent the same area of interest, has enjoyed a much greater following in the United States and in other European countries and has little bearing on the Scandinavian approaches under consideration here. Unlike its Scandinavian counterpart, software engineering confines its attention to software as a product in its own right, largely divorcing any substantial examination of software requirements from the application context.

We shall examine this contrast more closely in the section on software development as design later. Let us now turn to the origin of ideas on work design to see how these two strands are intertwined.

Development of Ideas on Work Design

From the 1950s onward, beginning in Norway, considerable efforts were undertaken, making use of the sociotechnical systems approach, to promote pilot projects in a wide variety of companies in an effort to encourage industrial democracy. This participative approach has helped establish an essential basis for understanding what is important for human beings in their work. Its theoretical basis has been gradually refined and more closely tailored to the development of computer-based systems.

The values embodied in the sociotechnical approach are closely keyed to humanization. Project development proceeds on the assumption of a project-specific consensus, enabling all those affected by the project, including management, to be motivated to collaborate as if on a common mission. This assumption, termed harmony perspective, together with the sociotechnical-approach-related strategy of implementing its goals in a practical social context solely in the form of pilot projects, rather than building up a general strategy of democratization, was later sharply criticized by the trade unions.

Meanwhile, the trade unions were quick to recognize the increasing significance of computer technology for working life and began, in the 1960s, efforts to establish a position of their own with respect to the application of
computer technology in the interests of the working population. The trade unions, for their part, based their view on a conflict perspective, which they understood as part of a continued labor dispute, and gave precedence to promoting the general democratization of working life over individual pilot projects whose wide-scale effectiveness they questioned.

A crucial factor at this juncture was the cooperation between the unions and a group of computer scientists headed by Kristen Nygaard, one of the authors of SIMULA. Nygaard sees himself both as a scientist and a trade unionist and has shown great personal commitment and professional skill in promoting the evolution of an independent trade union approach to the development of computer-based systems.

This work was soon crowned with considerable success in the form of legislation and skeleton agreements. An essential feature of these efforts was the fact that its architects were not content simply to criticize the application of computer technology and encourage a better representation of workers' interests; they also evolved an independent and technologically innovative vision of how computer technology should be applied in the interests of the working population, which became known as the collective resource approach.

The alternative methodology propounded by the trade-union-oriented scientists was based essentially on consciously value-guided approaches to design, elaborating and employing innovative concepts of systems development and on the use of computer technology in organizations. Particular importance is also attached to the idea of training schemes. These are designed to familiarize workers with the new technologies, in order to safeguard and to improve their qualification levels.

A comparison between these two approaches to work design must be drawn on three different levels:

1. The more recent collective resource approach has taken up some of the ideas of the earlier sociotechnical approach and incorporated them into its own theoretical foundations.

2. What the later approach achieved in terms of critically appraising and further developing the methodological basis of the earlier approach has been mirrored to some extent by concurrent developments within the sociotechnical school itself.

3. The objectives pursued in both cases are related. The collective resource approach, however, gives much greater emphasis to the idea of democratization. There are, moreover, distinct differences between the respective strategies employed for implementing the goals in a practical social context: The earlier approach relies here exclusively on internal pilot projects conducted within the individual organizations; the other adopts an independent position that the trade unions have been instrumental in developing, advocating, and successfully establishing.
Historical Shifts

As far as the technical aspects of the work are concerned, similar gradual shifts in emphasis are perceptible in all of the approaches:

- Like everywhere else, there has been a shift in technical trends, away from batch processing on mainframes to interactive systems implemented on modern computers with sophisticated access facilities. This has led to the design of interactive systems as tools tailored to users' needs. In addition, the use and adaptation of standard software as a basis for system development has become an increasingly important factor in systems design.

- The problem of embedding computers in organizations was originally tackled using a classical systems approach, as shown in more detail in the next sections. In contrast, concern with providing users with suitable tools has caused attention to be focused on work tasks, this being particularly evident in the collective resource approach, but not confined to it. More recently, a trend toward cooperative work processes has emerged, in which the computer is no longer seen as a tool for use by individuals, but rather as a communication medium.

- System development is being increasingly considered integrally as a learning process for individuals, or as interlocking learning processes involving several individuals. Here, the attempt is made to find out empirically what software development is. Methods are ranked according to their utility for intelligent action in a given situation. This implies a critical dissociation from the internationally prevalent ideas of software engineering.

- Cooperation between developers and users is considered a crucial factor and is given methodological support. User participation was a demand already recognized by the sociotechnical and other system-oriented approaches. It was, nevertheless, later criticized as being inadequate, particularly by adherents of the collective resource approach, because it failed to provide sufficient codetermination rights for users and also ignored the issue of user qualification for the participation process. Today, precedence is given to mutual learning with guaranteed rights of codetermination, also covering conflicts between different interest groups. Special training schemes have been developed for this purpose. Various forms of prototyping are used to provide technical support for the process of mutual learning, to help progressively qualify users, and to adapt software to meet the needs of specific user communities.
These innovative approaches to systems development are now by no means being promoted by the trade-union-oriented scientists alone. They are also advocated by representatives of the sociotechnical school and by scientists pursuing an independent line and calling for a user-oriented approach in computer science.

The Systems Perspective—and Beyond . . .

Parallel to the development of the sociotechnical approach, there emerged in Scandinavia and elsewhere methodologies and languages for system description, design, and simulation. Here, the computer assumed different roles, appearing either as a component of the systems under design or as an aid to system simulation. For the purposes of this study, the former is of particular interest. Socially oriented technology design involves assessing the merits and limitations of system-oriented approaches as a conceptual foundation for software development and for embedding computers in organizations.

Viewed from what was eventually called the systems perspective, embedding a computer system in an organization means that the computer appears there as a component, interacting with other components (as a rule, human beings) to form a system, or subsystem of a larger system, fulfilling a particular function.

The process of developing computer-based systems, viewed from a systems perspective, means that the interaction of all system components must be conceptualized, defined, and, in the case of functions that are to be assumed by the computer, implemented. The problem encountered here is the relationship between the activities performed by individual human beings, acting as part of the system, and the interaction of the individual components of the system as a whole.

Existing system-oriented approaches differ from one another in several respects:

- They use the term *system* with different connotations.
- They define different subareas of reality as their frame of reference, this being the relevant point of departure in developing software and that to which the subsequent embedding of the computer system is related.
- They take different views of the way human beings work, as compared with machines.
- They embody different values and ideas about the way in which human beings and machines should interact in computer-based systems, and about which interests the system should serve.
The system-oriented approaches developed in Scandinavia embody, to a high degree, the values concerned with humanization and democratization that are of particular relevance here, in comparison with other approaches developed in an international context (cf. Mathiassen's critique of Yourdon's "hard" systems approach and Checkland's "soft" systems approach discussed in Section 6.2). Furthermore, the Scandinavian approaches often only provide for informal guidance to practitioners in system development, whereas the notions employed in the other approaches no longer appear adequate now. To some extent this can be explained by the long historical development of these approaches, going back to the pioneering work of Langefors and taking the notion of systems through several stages.

Langefors originally worked on strictly technical systems and later moved on to consider information systems in organizations. He understands a system in general to mean "... a collection of objects, called parts, which are correlated in some way" (Langefors, 1973, p. 35). An information system is a subsystem consisting of "... information sets needed for decision and signalling in a larger system ... containing subsystems for collecting, storing, processing, distributing information sets" (Langefors, 1973, p. 195). Unlike data, information is understood to mean "... any kind of knowledge ... that can be used to improve, or make possible a decision or action" (Langefors, 1973, p. 319).

Although this conceptual framework fails to take account of the position of individual human beings in the system, Langefors does stress the fact that taking individual design decisions in a suitable order is a crucial factor in developing information systems. To ensure that sufficient attention is paid to user needs, it would be necessary to take into account, first of all, factors relating to the overall user or use context, postponing consideration of technical aspects as long as possible. This view is in stark contrast to other system-oriented approaches, as well as to the product-oriented software engineering approach outlined and criticized later in this section. Langefors's ideas, then, if applied in the manner suggested by him, provide considerable scope for involving users in the system development process. This was the starting point for participative systems development in ISAC (see Section 5.1).

Nygaard, together with several of his colleagues, went a great deal further than Langefors by developing concepts for systematic work in system development and a system description language DELTA, based on SIMULA, suitable for computer-independent application in the early stages of development. As far as we are aware, there has not been a great deal of progress in the practical application of DELTA. Nevertheless, the basic notions concerning systems and their development embodied in this language are outlined here, because they play an important part in the overall view taken of computer science by the trade unionist school.
The authors of the DELTA report take the term system to mean "... a part of the world which we choose to regard as a whole, separated from the rest of the world during some period of consideration, a whole which we choose to consider as containing a collection of components, each characterized by a selected set of associated data items and patterns, and by actions which may involve itself and other components" (Holbæk-Hanssen, Håndlykken, & Nygaard, 1975, p. 15).

What is new and striking here, in comparison with Langefors's definition cited earlier, is the element of choice or decision ("we choose to") in defining the limits of the system and in selecting its components. The same part of reality may be viewed as a system in different ways for different purposes.

It is obvious that this definition of a system is firmly rooted in the object-oriented thinking of the programming language SIMULA. Although this way of modelling reality is extremely powerful, it is not equally suitable for modelling all subareas of reality. If we go by this definition, we must, for the purposes of embedding computer systems in organizations, describe both human beings and computers as components. When applied to human beings, this has an artificial ring about it, and, when applied to computer systems, its limitations become apparent when we go from batch-processing systems to interactive systems.

This is the reason why Nygaard's notion of system, too, is today no longer adequate for discussing the human-oriented embedding of computer systems in organizations. The notion of component requires modification so as to enable fluid entities such as the interlocking work and communication activities of users to be regarded as elements, and their interlacement with computer functions to be considered. This would imply giving up attempts to place human beings and computers within the same conceptual framework (see also Floyd, 1987).

As seen from now, it would appear more fruitful to develop conceptual frameworks for considering the interpretation of computers in their social context. This must be accomplished by considering human activity in its own right and viewing the role of the computer in this context. A notion of social systems could then still be used, but it would have to refer to human activities as its elements. The more common alternative in Scandinavia is to abandon the system perspective altogether, and to use other concepts for describing the relationship between human beings and computers.

Software Development as Design

This section is devoted to contrasting views of software development: the production view inherent in software engineering and the design view adopted explicitly or implicitly in the work of Scandinavian researchers (in particular in Denmark) that is compatible with innovative forms of system development.

To make this section more readily intelligible, we feel it necessary to outline
briefly the basic trends in programming methodology and, by extension, software engineering as it has evolved since 1968, the year software engineering emerged as a research program at its founding conference in Garmisch-Partenkirchen. Naur, in fact, was one of the editors of the report for this conference. Software is seen here as a product in its own right to be manufactured on the basis of fixed requirements. Software development is compared with other industrial forms of production, and a similarly transparent production process, independent of the individual developer, is called for. Basic assumptions behind this paradigm are:

- Software development is based on fixed requirements. These can be either determined independently of the application context or obtained by analysis of the application context using formalizable information-processing techniques (different instances of a system perspective).

- The development of software products should be divided into temporal phases. Each phase results in a document that is considered as both an intermediate product and as a basis for the next development phase.

- Communication about software products should, as much as possible, take place exclusively on the basis of defining documents. These must be complete, consistent, and unambiguous, and they must define what the program is to do without stating how it is to do it.

- Methods take the form of rules for work, which specify the means of representation to be used and prescribe the work steps to be performed and the sequence in which this is to be done (e.g., top-down design, starting from a defining document that specifies the overall solution in an abstract manner, proceeding in a number of refinement or concretion steps until a complete formulation of the program is achieved).

- The most important quality criterion is the so-called correctness of programs. Correctness means that programs function in conformity with the defining document, without any reference being made back to the application context. Correctness is achieved by using mathematical procedures and formal proofs based on these.

- Defining documents should, as much as possible, be written in formal language, natural language being unreliable because of its ambiguity. Reasoning in ordinary prose is, therefore, less meaningful than formal models. Human comprehension takes place on the basis of rules to be employed for the unambiguous interpretation of formal models.
• Human beings are assigned subtasks in software development on the basis of the product structure that is derivable from the defining documents. It is their job to develop programs to correctly solve the subproblem assigned to them. The work style of human beings should, as far as possible, be rule governed, not differing essentially from that of machines. Wherever technically feasible, subtasks in software development should be carried out by the computer.

Of course, every software engineering author will openly admit that real software development in individual project situations deviates from this ideal. And yet prospective computer scientists are still trained predominantly in accordance with this paradigm. It does, however, provide a poor methodological basis for socially oriented technology design, because no support is given to communication and learning processes between developers and users. Also, no attention can be paid to the problems involved in suitably embedding the software in the application context, and insufficient consideration is given to the computer as a tool for problem solving (see also Floyd, 1987).

It should be noted that the software engineering tradition, outlined previously, is based on conceptions about what software development should be. In other words, specific rules of action are imposed in advance on the actual development situation, rules that, according to the assumptions behind this paradigm, result in the production of high quality software.

In contrast, Scandinavian approaches are based on software development as it really is (i.e., they proceed from experiments on real program development processes and empirical studies of real-life project situations). They take account of the problems and tasks encountered here in their totality, and they view methods as aids for the benefit of workers, enabling them to behave intelligently in the development situation and use their skills to produce high quality software.

As early as 1974, Naur (1974, p. 296) called programming an “activity of overall design with an experimental attitude.” Design, here, refers to the product software, its embedding in its social context and the development process. Thus, the general view now is that the system development process has considerable influence on the resulting product. This is the reason for the keen interest in methods for system development.

Moreover, increasing consideration is being given in program development to (a) the social and communicative structure of organizations and (b) the cooperative work processes to be supported. Here, a conscious effort is being made to combine insights from several different scientific disciplines (e.g., organization theory, philosophy, psychology, and linguistics).

A coherent approach to systems development is being pursued by Lars Mathiassen, now professor at Aalborg University, formerly associated with the system development group at Århus University. This group belongs to
the trade unionist school headed by Kristen Nygaard. Basing their work on Mathiassen's doctoral thesis on system development methods (Mathiassen, 1982), they are engaged in research on the development of computer-based systems in teams. Their interest focuses primarily on meaningful forms of cooperation between members of development teams and between developers and users, their aims being to manufacture products of high technical quality and to design work environments that take account of user needs.

Opposing the prevalent system perspective on the use of computers, Mathiassen advocates a tool perspective and a situation-specific attitude to software development. The major features of his approach—identifying process-related activities, emphasizing their correlation with product-related activities, and tailoring the methodology to support the work, communication and learning processes taking place during system development—constitute, in our view, a significant contribution to our understanding of the development and design of computer-based systems. These processes rely, in particular, on cooperation between developers and users.

From the outset, the interest-governed trade unionist line of research and development saw the workers as playing a crucial role in the development, installation, and use of computer technology. This led to a redefinition of the relationship between developers and users. In the course of a number of development projects (see Section 8) various cooperation concepts were tested, and traditional participation approaches were extended by adoption of the following principles:

- Mutual learning: Users and developers alike are reliant on a mutual process of learning and communicating.
- Designing by doing: Early experimentation and testing, such as using prototyping and promoting communication and learning processes.
- Interest-governed codetermination by the trade unions in the conception, design, and control of computer-based work processes.

As a technical procedure for supporting mutual learning, the technique of prototyping has recently been adopted by numerous development projects in Scandinavia.

Prototyping is essentially concerned with early implementation of selected parts of the desired software system that are designed to illustrate aspects of particular interest. The Scandinavians were quick to recognize the benefits of systematically using prototyping for development strategies that serve to promote cooperative communication and learning and work processes and that aim at designing computer-based systems with a view to improving work quality and user qualification and establishing democracy at the workplace.

All of these methodological considerations may be seen as embodying a
chiefly process-oriented approach. This goes beyond the mainly product-oriented view of software engineering and induces us to reappraise (a) the role of professional software developers, (b) the importance of desirable forms of cooperation and the division of labor, and (c) the significance of methods and supporting tools in software development.

Controversies and Agreement in Scientific Practice

The previous account is not intended to mislead the reader into believing that there is general agreement on the merits of these approaches in Scandinavia. In fact, we are aware of several controversies and disagreements. The trade-union-oriented scientists accuse the sociotechnical school of failing to recognize the fact that the workers' interests can only be safeguarded by the trade unions. Advocates of the tool perspective on the use of technology criticize the rivalling system perspective, which underlies the thinking in the whole field of computer science, for failing to support the design of computer-based systems geared to the needs of human users. By the same token, many other computer professionals consider cooperation between scientists and trade unions to be problematic, because, in their view, mixing scientific work with trade union goals leads to a conflict of priorities between scientific insights and the pursuit of workers' interests, which is a handicap to in-depth scientific work.

These disagreements are argued out and discussed at conferences and at Scandinavian centers of research. There are compatible and incompatible views as well as research milieus. To us, the lively climate for such discussions seems to be very much in keeping with the spirit of the Scandinavian Approach. Perspectives, opinions, and values are brought out into the open. They enter into the argumentation and thus lead to deeper insights, both in system development projects and in the scientific discussion as a whole.

4. ORGANIZATION-ORIENTED AND WORK-ORIENTED APPROACHES

4.1. The Sociotechnical Systems Approach

The emergence in the Scandinavian countries of a technology and work design ethos geared to humanization and democratization was strongly influenced by the sociotechnical research tradition rooted in organization theory.

Established in Britain in the years following World War II at the Tavistock Institute of Human Relations by the research group around Eric Trist (Mumford, 1987), the sociotechnical approach was taken up in the 1960s by the group around Einar Thorsrud at the Norwegian Institute for Social
Studies and was tested and further developed in the course of a number of practical projects on industrial democracy. From there, the sociotechnical approach spread to other Scandinavian countries, especially Sweden. During the 1970s, it was further developed by the research group around Enid Mumford at the Manchester Business School who, together with the Scandinavians (i.e., Hedberg and Bjørn-Andersen), elaborated it with an emphasis on computer applications. The sociotechnical approach plays a crucial role in current research and development work on computer-based systems throughout Scandinavia.

The basic idea behind the sociotechnical approach is to view business organizations, such as industrial enterprises and other organizations, within the production, administration, and service sectors as systems consisting of technical and social subsystems. These are related to one another and to their environment in a variety of ways, and are therefore termed open systems.

The notion of open systems is borrowed from the school of organismic biology concerned with systems theory. Here, biological organisms are seen as living systems, and these living systems are described as being open. Open systems are models of identifiable entities, engaged in a necessary process of exchange of materials with their environment and possessing the ability to resist the continual changes acting on them, whether these come from inside or outside, by constantly re-establishing a steady state so as to ensure that at least the vital exchange of materials with the environment can be adequately performed. By analogy with this view, the sociotechnical systems approach assumes that a business organization is an open system and that the technical and social subsystems that constitute it, which are governed by their own intrinsic laws, have to be matched to one another and to their environment if the organization is to enjoy lasting stability.

Another feature of the sociotechnical line of research, besides its reliance on systems theory, is the fact that it does not confine itself to consideration of the technical and structural aspects of work within a particular sphere of activity when analyzing the internal structure of business organizations and industrial work environments. Rather, it focuses attention on the individuals acting in the work process (i.e., human beings). Factors such as the well-being and contentment of the individual worker, together with scope for self-determination and personal development, are considered just as crucial as the effects of technological, organizational, and structural changes on social groups as far as work productivity and the stability of the sociotechnical systems are concerned.

The central paradigm of the sociotechnical approach to work organization consists of (a) the postulation of a causal relationship between job satisfaction and work productivity and (b) the recognition that technology must be compatible with organizational and social needs, if both of these are to be increased. This takes account of the fact that "... if the technical system is
optimized at the expense of the human system, the results obtained will be sub-optimal. The goal must therefore be joint optimization of the technical and social systems" (Mumford, 1987, p. 63).

The specific contributions made by the founders of the sociotechnical systems school to improving our understanding of work organization and its design are described by Mumford (1987) as:

- elaborating the concept of the sociotechnical system,
- viewing an organization as an open system,
- establishing the principle of organizational choice—observing the need to match social and technical systems in the most appropriate way,
- recognizing the importance of autonomy and self-determination for social groups,
- better understanding the problems of work alienation and determining their importance as a factor.

The sociotechnical systems approach focused from the outset on the organization of work and not on analyzing the general socioeconomic conditions affecting it. However, the fact that the founders of this approach turned their backs on a scientific tradition of work organization geared exclusively to profitability—choosing, instead, to emphasize the importance of the worker not only as a production factor but also as a human being, and focusing on factors such as job satisfaction, autonomy, and self-determination for social groups, scope for personal development, and so on—distinguished the sociotechnical approach from other approaches to work organization current in the 1940s and 1950s.

This fact, too, must be considered the reason for the adoption of the sociotechnical systems approach by the group around Einar Thorsrud at the Norwegian Institute for Social Research when, in the late 1950s, they were commissioned by the Norwegian government to explore suitable social strategies for solving increasingly urgent structural problems in the economic and social sectors. Ehn and Kyng (1987) asserted that:

At the end of the fifties the growth in many areas seemed to come close to the limits set by the natural resources. Differently with the resources which depend on human initiative and creativity. . . . Both economic considerations . . . and a growing unrest in the left wing labour movement contributed to the creation of strong interests." (p. 23)

For reasons related to a wide variety of sociopolitical, historical, and sociocultural factors that we are unable to discuss in detail here, the problems
of economic growth and work productivity, technological innovation, and structural changes in the socioeconomic sector were, first in Norway and later in the other Scandinavian countries, tied up from the start with the issue of industrial democracy.

Under a Norwegian government scheme to promote industrial democracy (Ehn & Kyng, 1985b), in which trade unions and employers' associations were also invited to participate, the Norwegian Institute for Social Research was commissioned in the early 1960s to conduct several industrial research projects (Emery & Thorsrud, 1976; Emery & Trist, 1969). This was the first time the sociotechnical systems approach was applied in a Scandinavian context and, with the cooperation of the Tavistock Institute of Human Relations, it was tested and elaborated in several government-funded projects.

Focusing on the issues connected with industrial democracy meant that industrial research projects dealing with technological innovation in Norway were necessarily concerned with worker participation in the restructuring of industrial work processes. This involved asking questions such as: Under what conditions can more rights and responsibilities be achieved for the individual in the workplace? Which organizational models enable work groups to exercise more personal control over their work environment? How are the new technologies to be embedded in the existing organizational structure so as to increase both work quality and productivity, while at the same time ensuring greater job satisfaction? (Mumford, 1987). These questions were followed up by researchers in the course of numerous experiments conducted first in Norwegian companies and then, from the mid-1960s onward, in Swedish companies as well. This led to the development of a wide variety of concepts, models, and techniques for participative system design and worker control over the work environment.

Toward the end of the 1960s, critical voices were raised, again in Norway, where the trade unions took issue with advocates of the sociotechnical systems approach over its confinement to industrial organizations. The elaboration of the sociotechnical systems approach to work design, inherent in the approach itself, and in particular its application to the design of computer-based systems, were goals pursued during the same period by the research group headed by Enid Mumford at the Manchester Business School, in close cooperation with Bo Hedberg and, later, Niels Bjørn-Andersen (Bjørn-Andersen, Hedberg, Mercer, Mumford, & Sole, 1979; Hedberg & Mumford, 1975).

Mumford also played a crucial role in spreading this approach throughout Scandinavia during the 1970s and right up to the present day (Bjørn-Andersen & Hedberg, 1977; Mumford, 1976, 1978, 1983a, 1983b, 1985; Mumford & Sackmann, 1975). On the basis of experience gained in the so-called "participation projects" on industrial democracy conducted in
Scandinavia during the 1960s and the considerable know-how acquired in the course of her extensive consultant and training activities in conjunction with the introduction of computer technology, she succeeded in developing concrete methods for participative systems analysis, design, and evaluation. Moreover, by publishing extensively on the subject, she helped widely disseminate the participative sociotechnical systems approach, particularly in computer application fields.

The underlying concept of participative system design postulated by Mumford and others is based on viewing computer applications as sociotechnical systems with multiple goals that are assessed differently by different groups according to their respective viewpoints. These different goals may be pursued and attained jointly if the values underlying them are made as explicit as possible from the start. This basic assumption, which has found its way into the literature under the name of the harmony thesis, is based on the notion that each organization (more specifically, each industrial organization) is governed by a common overriding goal or "mission" (Mumford, 1985; Mumford, Land, & Hawgood, 1978; Mumford & Weir, 1979). Existing conflicts of goals between different groups or individuals must be identified, and these can, by structural changes in the organization, be settled in such a way that the stability of the sociotechnical system and its effectiveness with respect to the common goal are restored.

It is precisely these assumptions of the sociotechnical systems approach that have given rise to widespread criticism of it by the trade unions. This, in turn, has led to the emergence of an interest-governed trade-union-oriented line of research and development.

Both the positive and the critical reception of the sociotechnical systems approach have substantially shaped Scandinavian approaches to the design of computer-based work processes. The sociotechnical systems approach has won a firm place throughout Scandinavia in applied computer science research, as well as in the practical development of computer-based systems and their embedding in organizations. It is being systematically tested and elaborated in numerous research and development projects (Bjørn-Andersen, 1980, 1985; Bjørn-Andersen & Kumar, 1985; Hedberg, 1980; Höyer, 1980).

In particular, Bjørn-Andersen in Denmark, Hedberg in Sweden, and Höyer in Norway, through the numerous research and development projects they have conducted in recent years, have been instrumental in further developing the approach of the sociotechnical systems school.

4.2. The Collective Resource Approach to Computer-Based Work Design

After critical appraisal of the experience gained from the participation projects conducted during the 1960s, the trade unions came to the conclusion
that the research and development strategies pursued by government research institutions were not conducive to a qualitative improvement in the position of workers and their organizations in the business and industrial sectors and were consequently bound to fall short of their goal of industrial democratization.

From the beginning, the Swedish Trade Union Federation took a skeptical view of the industrial participation projects that were, for the most part, launched and coordinated by the Swedish Employers' Association (SAF) in the late 1960s along the lines of the sociotechnical systems approach. Their criticism was levelled at the strategy, pursued in those projects, of concentrating on the interests of the individual worker, thus hindering, in their view, the development of collective solutions acceptable to the trade union movement as a whole (Swedish Trade Union Federation, 1977).

In Norway, where the Trade Union Federation was directly involved in the government-funded participation projects conducted in various areas of industry, a revision of this strategy was called for in the late 1960s, above all by the industrial trade unions. Analyzing the experience gained from the industrial projects conducted within its own organizational sphere, the Iron and Metal Workers' Union (NJMF) showed that reliance on the sociotechnical approach as a theoretical model was diverting attention away from the need for changes in social and economic policy, and consequently from the social conflicts and conflicts of interest which these necessarily entail (conflict thesis). The participation projects had, admittedly, shown that it is, in principle, possible to develop new and better working conditions for individual workers and groups of workers, "... but if we neglect to make initiatives of this sort an integral part of the overall endeavour, they will be of no help in establishing a truly democratic situation..." (Nygaard & Bergo, 1973, p. 171). The trade unions attributed the workers' gradual sense of resignation and their waning interest in continuing the participation projects to the fact that it was impossible to correlate the experimental organization models and participation concepts with the interests of the workers on the shop floor, which were dictated by their everyday working situation and their status in the company as a whole.

In the trade unions' view, a process of democratization in the interest of the workers had to entail a change in existing power structures and hierarchies (i.e., a change in the distribution of power, both at the shop-floor level and in society as a whole). And they concluded that what was needed was an interest-related strategy, taking into account the specific historical, sociopolitical, and sociocultural conditions in Scandinavia, that would enable the trade unions to play an independent part in the process of democratization.

In addition to initiating legislative measures that, once passed, helped extend workers' co-determination rights at company level and create a new programmatic basis for tackling the issue of industrial democracy in all
spheres of society, Norway's trade unions also launched research and development activities of their own.

The crucial importance of planning, controlling, and data processing for bringing about structural changes in the economic sector was recognized by the late 1960s. However, the trade unions lacked the necessary basic know-how in these fields, they were without appropriate education and training schemes, and they lacked a clear sense of direction that might have helped them activate the workers and strengthen their position. Consequently, the establishment of an interest-governed, trade-union-oriented research and development policy in these areas was seen as being imperative for developing an independent trade unionist strategy on technological and labor issues.

The principal ideas behind the new research and development activities, launched by the Norwegian trade unions in the early 1970s and taken up by researchers at universities and other government institutions, were:

- to redefine the role of the trade unions in planning and introducing new technologies, and in the redesign of the company work processes and their organization which these entail;

- to activate trade union groups at company level in order to safeguard their interests in the conception, design, and application of new technologies;

- to launch and supervise their own research and development projects for acquiring and distributing know-how at local and central trade union level;

- to set up their own interest-governed, action-oriented training and education schemes that would enable workers at all levels to bring their influence to bear on the redesign of work processes, and to press for democratization.

The new theoretical foundations, strategic models, methodologies, and concepts for this work were, for their part, developed in the context of national, government-funded application projects that were launched by the trade unions and conducted in collaboration with researchers at several different institutions. Of the numerous research projects (beginning in Norway, and then, from the mid-1970s onward, spreading to Sweden and Denmark) that were instrumental in establishing a trade unionist line of research in computer science and aimed at developing strategies for democratization, we have singled out three that may be considered exemplary and that have since attracted considerable attention in the literature of both the German-speaking and English-speaking scientific communities: NJMF (Nor-
way), DEMOS (Sweden), and DUE (Denmark; Bjerknes et al., 1987; Bjørn-Andersen, 1982; Briefs, Ciborra, & Schneider, 1983; Ehn & Kyng, 1985a; Kubicek, 1979; Kyng & Mathiassen, 1982; Sandberg, 1979).

NJMF—Norsk Jern og MetallarbeiderForbund

The Planning, Control, and Data Processing Project, launched by the Norwegian Iron and Metal Workers’ Union in 1971 and revised and conducted in collaboration with the Norwegian Computing Center under the direction of Nygaard and Bergo, undoubtedly paved the way for the development of the Scandinavian Approach to computer science, an approach relying explicitly on the trade unions as major champions of the issue of industrial democracy.

The results of this project, which ran for a period of 2 years, included the elaboration of strategic positions; the conclusion of industrial agreements on the introduction and use of data processing (Kubicek, 1979; Nygaard, 1987); the production of readers on planning, controlling, and data processing (Nygaard & Bergo, 1973); and, of course, the project report itself (Nygaard & Bergo, 1975). All of these formed the basis for numerous subsequent projects conducted by Scandinavian scientists and trade unions, and they were made known to the international scientific community thanks especially to the numerous lectures and publications of Kristen Nygaard (Nygaard, 1977, 1979; Nygaard & Bergo, 1975).

DEMOS

In 1975, the Democratic Control and Planning in Working Life: On Computers, Trade Unions, and Industrial Democracy Project (DEMOS; Perby, 1977) was launched in Sweden by the Central Federation of Industrial Trade Unions (LO) and conducted in collaboration with an interdisciplinary research group that was transferred to the Swedish Center for Working Life when it was created 2 years later. Devised along the same lines as the NJMF Project, the DEMOS Project was, in addition, concerned with the development of new negotiation models (see Figure 2). The idea was to develop a strategy that would secure independent trade union research work at the company level and worker participation in management project groups, as well as a say in problem solving and decision making (Carlsson, 1977; Carlsson, Ehn, Erlander, Perby, & Sandberg, 1978; Ehn & Sandberg, 1979, 1983; Sandberg, 1983).

DUE—Demokrati, Udvikling og EDB

In 1977, the Danish Central Federation of Trade Unions (LO) submitted an application to the Danish National Agency of Technology for funding of the Project on Democracy, Development, and Data Processing (DUE). The
project was duly approved and subsequently conducted in collaboration with a group of junior scientists and computer science students under the supervision of Lars Mathiassen and Morten Kyng at Århus University.

The groundwork for this project was laid by Nygaard's critique of the findings of the NJMF Project, presented during a series of lectures he gave as a visiting professor at Århus University, and by a cooperative project with the local trade unions launched in 1976 as a result of Nygaard's work, which was concerned with training shop stewards.

The DUE Project followed its Norwegian forerunner in both its objectives and its organizational structure. Unlike its predecessor, though, it focused on determining the effects of computer application on working conditions, as seen by shop stewards, in business organizations and in public institutions. Its findings were to be used to develop, together with the shop stewards and local
trade union bodies, strategies for the design of computer-based systems at the local company level, thus laying the foundations for general industrial strategies to be adopted by the central trade union organizations.

In the first phase of the project, a broadly based survey was carried out with the help of local trade union bodies in 165 different work environments throughout Denmark. Of the 96 institutions that returned the questionnaires, 12 were selected for further investigation.

In the second phase, detailed investigations of the problems in three specific workplaces were conducted. Resources were developed within the local trade unions to increase their influence on the use of computer technology; a study was made of the system design methods available in computer science; suitable alternatives were developed to eliminate the problems identified; and the task of compiling new educational and training material for trade unions and universities was embarked upon.

Finally, in the third phase, the theoretical findings and practical know-how acquired in the previous phases were used in producing instructional material for trade union training courses and compiling empirical studies. The objective here was to make available the findings obtained in the various trade union sectors and to initiate discussions throughout the trade union movement on the new technologies and, more particularly, on technology agreements at the company level or for a whole industrial sector.

"Second Generation" Projects

Evaluating the experience gained during the 1970s, Ehn and Kyng (1987) came to the following conclusions:

For the last decade the ideas, working practices etc. of the first collective resource projects have spread throughout Scandinavia. Local data agreements have been negotiated, data shop stewards appointed, union clubs have formed their own investigative groups, and the unions have arranged numerous courses to support the clubs in their attempts to influence the design and use of computer systems.

But although growing, the extent and impact of these activities did not meet the initial expectations. It seemed that one could only influence the introduction of the technology, the training, and the organization of work to a certain degree. From a union perspective, important aspects like opportunity to further develop skill and increase influence on work organization were limited. Societal constraints, especially concerning power and resources, had been underestimated, and in addition the existing technology constituted significant limits to the feasibility of finding alternative local solutions which were desirable from a trade union perspective.
As an attempt to broaden the scope of the available technology it was decided to try to supplement the existing elements of the collective resource strategy with union based efforts to *design* new technology. The main idea of the first projects, support to “democratic planning,” was complemented by the idea of designing “tools for skilled work.” (p. 32)

The first and most important project on direct technology design was the **UTOPIA** Project, begun in 1981, which we take a closer look at in Section 8.3. The project was formed by a cooperative effort between the Nordic Graphics Workers’ Union and research institutions in Sweden and Denmark. The aim was to design computer support and professional education for integrated text and image processing in the newspaper industry. Here, the attempt was made to base design on specific principles. These included (a) quality of work and products, (b) democracy at work, and (c) education for local development. The objective was to explore the prospects for a technology design consistent with workers’ interests.

The project, which was concluded in 1986 after a 4-year period,

... contributed to the development of alternative participative and skill-based design techniques in general, as well as to more skill-based and democratic work organization in the newspaper industry, to computer-based tools which support such work organization, and to professional education for printers. (Ehn & Kyng, 1987, pp. 32–33)

The research and development methods evolved in the course of the previously mentioned projects and a number of others* during the 1970s—methods whose elaboration owed a great deal to the trade unions and researchers involved—were embodied in a distinct approach to computer-based systems design that won recognition throughout Scandinavia. This interest-governed, trade-union-oriented approach, known as the Collective Resource Approach to Systems Design, emerged as a result of a process of interaction between theory and method development by computer scientists

---

* Other projects within or related to this Scandinavian tradition are:

- The **Dairy Project** that was conducted by architects but used methods and perspectives similar to those employed in the **DEMONS** and **DUE** projects (see Steen & Ullmark, 1982).
- The **PAAS Project** that, besides contributing to a theoretical understanding of changes in skills when computers are used, also developed methods for trade union design work (see Göranzon, 1984).
- The **Bank Project** that was conducted by researchers originally from the sociotechnical tradition but who worked closely with trade unions and with methods and perspectives very similar to those used in the collective resource projects (see Hedberg & Mehlmann, 1983).
- The **Carpentry Shop Project** that worked with methods and a design perspective similar to the **UTOPIA** project but within a “low tech” area (see Sjögren, 1979–1983).
and implementation thereof in terms of practical trade union policy. It was precisely this process of interaction on which theoreticians focused their attention and which practitioners adopted as their guiding principle.

The collective resource approach to research and development in the field of computer science must be seen as part of a strategy for promoting industrial democracy, and to this extent it must be considered a genuinely Scandinavian approach. The sociotechnical systems approach, in common with other theoretical and methodological approaches to system development widely used in Scandinavia, is employed elsewhere, too (Bjørn-Andersen & Skousen, 1984). In contrast, the development of a line of research pledged to trade union interests and aimed at industrial democracy—an undertaking carried out at universities and other research institutions in conjunction with government agencies and employers' associations—already constitutes, in its present form, a specific sociopolitical approach to tackling issues in science and technology that distinguishes Scandinavian countries from others of the Western world.

For this reason, we attempt to briefly outline in the following section the basic theoretical and methodological assumptions behind the collective resource approach. We do so by reference to a recent article by Ehn and Kyng (1987) entitled "The Collective Resource Approach to Systems Design."

**Basic Assumptions Behind the Collective Resource Approach to Systems Design in Scandinavia**

An essential element in the collective resource approach to systems design is its analysis of the design and use of computer-based systems in Scandinavia today, interpreted in the light of a general notion of societal work. From this notion are derived the organizational structures in which concrete labor processes are executed.

Labor processes are processes to produce "use values," tangible and informational products or services of some kind. Use values are produced by people using various instruments of work (tools, machines, methods, and techniques) to refine objects of work (raw materials, ideas, and existing services). Conception and execution are fundamental parts of people's work in this process.

The division of labor between different groups of people, their respective qualifications to accomplish different tasks, and the quality of the use values being produced are key aspects of the labor process in the context of work democratization.

The design process performed when developing computer-based systems reflects, in principle, the division of labor into conception and execution. In the design process the use process is conceptualized; the things designed are technology, use values, work organization, and skill requirements. In the use process, work is executed given the constraints and opportunities set by the
design process. Conversely, the characteristics of a given use process also set constraints and opportunities for the design process. It follows that design and use processes should be seen as two separate labor processes and, at the same time, as parts of a total labor process.

One of the basic ideas behind the collective resource approach is that:

. . . theses on changes of labour processes must be applied to the totality as well as to the separate parts, to the division of labour between different groups of workers within the labour process of systems design, to the division of labour between systems design and the use labour process, and to division of labour within the use labour process. (Ehn & Kyng, 1987, p. 35)

Design of computer support is design of (conditions for) labor processes. Here, labor processes cannot be reduced to information processes. The design of computer support for labor processes calls for professional experience and know-how, both with respect to the labor process in which the computer is to be used and to the field of computer technology and its design. The design of computer support for labor processes must therefore be carried out with the users; it cannot be performed either for or by them.

In democratizing the design and use of new technologies in Scandinavia, the trade unions have assumed an active role, particularly at the local level. Clear identification of the differences or conflicts of interest between representatives of capital (management) and workers (trade unions) and determination of the resulting differences between their respective demands on technology and work design should not be seen as inhibitions, but rather as prerequisites to cooperation on technological and economic changes. An important prerequisite for trade union participation in management design processes is a parallel and independent process of accumulation of knowledge on the part of the unions. Independent trade union research and development is therefore seen as a necessary complement to traditional strategies of systems design.

Local union design efforts with respect to new technologies and structural changes have to be supplemented by central union design activities. In particular, local unions need external resources and support in their design activities.

Because the existing technology in many cases restricts the possibilities of locally attaining trade union objectives with respect to work quality and organization and skill, current interest in both scientific and trade union quarters is focusing on research into and development of new technologies. Suitable methods and concepts for technology design geared to improving the quality of work and work products and, at the same time, improving workers' skill levels would enable the trade unions to influence the supply of such
methods and concepts by specifically influencing demand for new technological and organizational solutions.

The trade unionist line of research attaches major importance to the training and education of those involved in the design and use of computer-based systems. An important element of the collective resource approach, on both a theoretical and practical level—and one to which computer scientists have made a significant contribution—is the production of educational material and the setting-up of training schemes (including courses of study) that deal with the issues of technology development and work organization from the trade union perspective.

The experience gained over the past 10 years in evolving an independent trade unionist line on work and technology design has resulted in a wide variety of new conceptual and methodological approaches, as well as producing a number of techniques and tools for supporting the development of computer-based systems (i.e., relating to computer science in a narrower sense). Some of these systems, which we feel may be fruitfully examined independently of their immediate Scandinavian setting, are examined later on in this study.

5. SYSTEM-ORIENTED APPROACHES

5.1. Development of Information Systems: The ISAC Methodology

In this section, we focus on a theoretical methodology for developing information systems, the foundations of which were laid by Langefors in the 1960s and earlier. The methodology was further elaborated by the ISAC research group headed by Mats Lundeberg, professor at the Stockholm Business School. It was developed independently of the sociotechnical systems approach, though the two approaches are mutually compatible.

In Sweden, concern with the theoretical and methodological foundations of information systems development can ultimately be traced back as far as 1943. This was the year when Langefors began to look at the problems of developing information systems. This resulted, in 1964, in his publishing a theory of information systems—the first of its kind in such coherent form—which he evolved from a general systems theory (Langefors, 1973). This, together with other publications of his, has been highly influential in Scandinavia.

Langefors's ideas were taken up by Lundeberg, Goldkuhl, and Nilsson of the ISAC research group in 1971, when they began work on a systematic approach to the development of information systems (Lundeberg, Goldkuhl, & Nilsson, 1981).
Description

ISAC is a method that starts by considering the needs, problems, and ideas of the users, proceeding ultimately to the specification of manual activities and computer programs. Here, the term method signifies a number of well-defined and coherent work steps, including rules as to what is to be documented during these work steps and how this is to be done. Before explaining the concepts behind the individual work steps, Lundeberg et al. (1981) defined the underlying notion of information systems as follows:

Information systems can be considered organized cooperation between people in order to process and convey information. . . . More precisely, an information system is a system that has been developed in order to create, collect, store, process, distribute, and interpret information sets. . . . Human beings create and interpret the information sets of an information system. . . . The parts of the activities that are automatable can also be performed by computer or other aids. (p. 10)

Information is taken to mean human knowledge. Data, on the other hand, represent information. A data system is, therefore, always (an automatable) part of an information system.

One of the fundamental concepts behind ISAC is its differentiation between user-oriented and computer-oriented activities. Langefors's aim has been to design computer-based information systems that provide adequate support for their users. He seeks to achieve this by calling on system developers to confine themselves initially to user- and use-oriented design decisions, postponing design decisions influenced by technical factors until as late a stage as possible. This concept is also embodied in ISAC, though here both types of activity were refined, as the methodology evolved, into a number of different work steps. Figure 3 illustrates the course of this development over a period of years.

A further refinement of work steps became necessary whenever practical testing of the method in a project situation showed that specific work steps were too complex. In such cases, too many goals were being pursued in a single step, and this was leading to unsatisfactory results.

Each of the work steps shown in Figure 3 pursues a specific goal. These are:

- Change analysis: Information systems are developed only when there is really a need for them.
- Activity studies: Information systems are developed only if they make positive contributions to the activities of the organization in some way.
Figure 3. Evolution of ISAC: A further refinement of work steps became necessary whenever practical testing of the method in a project situation showed that specific work steps were too complex. *Note.* From an ad hoc sketch by Mats Lundeberg during our interview with him, January 1987.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ED</td>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enterprise Diagnosis</td>
<td>Change Analysis</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>AS</td>
<td>AS</td>
<td>AS</td>
<td></td>
</tr>
<tr>
<td>Activity Studies</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IA</td>
<td>IA</td>
<td>IA</td>
<td></td>
</tr>
<tr>
<td>Information Analysis</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>user-oriented</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>computer-oriented</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSD</td>
</tr>
<tr>
<td>Data System Design</td>
</tr>
<tr>
<td>EA</td>
</tr>
<tr>
<td>Equipment Adaptation</td>
</tr>
</tbody>
</table>

- Information analysis: Information systems are developed in such a manner that the users understand what they contain and perform.

- Data system design: Information systems are developed in such a way that they are capable of change when needed and are not restricted to certain computer equipment.

- Equipment adaptation: Information systems are developed in such a way that they are effective and are adapted to the technical equipment used.

We now explain how the goals formulated for each of these work steps are practically implemented in terms of the methodology.

*Change Analysis.* What exactly are the problems experienced by the persons or groups affected? What interest groups are there in the problem areas identified? What goals are pursued by the different interest groups? Answers to these questions are obtained, first of all, by change analysis, and subsequently documented using the description techniques provided by the
methodology. Then, change alternatives are developed and evaluated together with the interest groups affected. At this stage, it is decided whether the problems are to be solved by developing an information system, or by other means outside the scope of ISAC.

**Enterprise Diagnosis.** The application of ISAC in development projects has shown that (a) change analysis is too time consuming, because too many change alternatives are discussed; and (b) there is a danger of choosing the wrong change alternative as a result of the selection criteria being too imprecise. In order to increase the effectiveness of change analysis, then, a further work step is introduced preceding change analysis, that is, enterprise diagnosis. The idea behind this is that once the aims of an enterprise as a whole have been determined, it is then possible to derive from these (and also for specific subareas of the enterprise) specific criteria for evaluating change alternatives. Enterprise aims may be of various kinds (e.g., increasing productivity, job satisfaction, or ergonomic design of work). Enterprise diagnosis is the subject of current research; no literature was available on it while we were working on the study.

**Activity Studies.** The decision to set about solving specific problems existing within an enterprise by developing an information system was taken in the previous work step. The important thing to define now is the limits of the future information system within the organization's activities in such a way that it really does solve the problems of the various interest groups involved. This should be done by reference to the users' needs, not to existing technical aids. To do this, the activities must be analyzed more closely and documented using the description techniques provided. Finally, the input and output information sets of the information system are defined, and the relevant properties of these information sets are described.

**Information Analysis.** The purpose of information analysis is to describe what the future information system, whose limits were defined in the preceding work step, is to contain and process. This description serves, first, as a means of communication for the interest groups affected in discussing the contents of the information system, and second, it constitutes an exact specification as a basis for the subsequent design of the data system. Information analysis starts from the results which the information system is expected to supply, and examines which processes are required in the information system in order to achieve these results.

**Data System Design.** The aim of this work step is to design an equipment-independent data system for the information system described in the previous work step. It is not until this stage that the decision is taken as to whether, and
if so which, parts of the information system are to be automated. For the manual parts of the system, the relevant work tasks are described; for the automated parts, the data structures and programs are designed. Particular consideration is given to the factors ease of maintenance, flexibility, and reliability.

**Equipment Adaptation.** The purpose of this final work step is to choose specific equipment and then to adapt the equipment-independent solution to this choice.

**Evaluation**

ISAC sets out to provide the following facilities in the development of information systems (cf. Lundeberg et al., 1981, pp. 6–9):

- The users can control the development of information systems by themselves.
- Information systems development can be performed in a number of small, manageable, and connected steps.
- Changes in and maintenance of the information systems can be carried out rapidly and safely.
- It is possible to achieve good cooperation between different professional categories in the development process (i.e., between users, problem-oriented systems analysts, and data-oriented systems designers).
- Development of information systems can become a natural part of the normal activities of the users.
- The problem-oriented systems analyst can be a catalyst (i.e., a person giving methods support to the users when needed).
- The data-oriented systems designers can concentrate on work with the data technical design.
- The information systems can be worked through thoroughly and can become well anchored in the organization.

ISAC is a comprehensive and universal approach, proceeding as it does step by step from problem definition to an operational computer system. It is also an open approach, because it clearly delimits the scope of its application beyond which it is necessary to work with other methods. And it is also an undogmatic approach, because it recognizes the most serious risk in working
with a methodology—the danger of applying a methodology mechanically without being aware of its pragmatic nature.

ISAC is a methodology that has been successfully applied. Besides being used throughout Scandinavia, it has been used mainly in Holland. In 1983, for example, 65 Swedish companies are known to have applied parts of the ISAC approach. The application areas included major projects by Saab-Scania, Volvo, and a number of government agencies.

So much for the appraisal of the ISAC methodology by its authors. In our opinion, ISAC is a rather conventional approach, viewing, as it does, the development of information systems from a result- or documentation-oriented perspective. Less conventional, however, is the way the methodology is applied in projects, in which a methodology is viewed from a wider perspective that takes into account the communication and learning processes of the people involved.

ISAC provides advice on how to support and to improve communication and learning processes during information systems development. However, in order to apply the methodology successfully, training under professional guidance is required, not only with respect to the individual work steps, but also for the communication processes involved.

5.2. Object-Oriented System Description

In the early 1970s, researchers at the Norwegian Computing Center developed a conceptual framework and a language for systems understanding and communication that may be seen as extensions of the concepts belonging to the programming language SIMULA (Birtwistle, Dahl, Myhrhaug, & Nygaard, 1973). The driving force behind this work was Nygaard, one of the authors of SIMULA and also the leading computer scientist of the trade unionist school. During this period, he also propounded his ideas in lectures he gave while serving as a visiting professor at Århus University. The basic concern of Nygaard and his colleagues was to use the concepts embodied in the language DELTA (Nygaard, 1976), along with a system description method attuned to these to create a tool for participative systems design that could be used in various kinds of communication situations. These include communication between systems analysts and various groups affected by systems development; communication on specific systems between trade union members confronted with the effects of such systems and systems analysts working with the trade unions; communication between computer experts and scientists from other disciplines; and communication between people working in interdisciplinary teams, in which the individual team members may have widely divergent ideas on how to set about designing a particular system.

In the project itself, efforts were made to adopt different views on systems
description and development. In a preliminary seminar held in October 1973, important ideas were presented by a number of leading scientists, among them Ole-Johan Dahl, Peter Naur, and the sociologist Stein Bråten.

One of the most important results of this project was its elaboration of a conceptual framework and terminology for systems, systems design, and systems description. The authors proceed from the basic assumption that, in today's world of ever increasing complexity, conscious efforts are being undertaken in a wide variety of spheres to integrate a large number of components in such a way that they interact purposively as a system.

Here, one and the same domain of reality may be seen as a system in different ways (e.g., as a social system, as an information system, as a decision system, or as a political system). Unlike other approaches, no attempt is made to reduce some of these systems to others (as in Langefors's decision system, e.g., which is seen as part of the information system). Instead, each of these systems is seen as corresponding to an independent viewpoint that must be considered on its own and in relation to other viewpoints in order to enable decisions to be made that take into account contrary views.

An essential aspect of this approach is that the authors make explicit their conscious decision to view something as a system with respect to a particular purpose. According to this view, systems do not exist as such; rather, we choose to look upon certain subareas of reality as systems (cf. the definition in Section 3). This emphasizes the importance of the perspective of those involved (i.e., the way in which they mentally structure the development situation on the basis of their experience, their expectations, or their personal values). Consequently, systems do not exist in themselves, but are dependent on an inquirer who seeks information about, or proposes decisions with respect to, the considered system.

A distinction is made here between mental systems, existing in the human mind and physically materialized in some way as states of our brain cells, and manifest systems external to human minds. The inquirer's task in system development is to develop models that are similar to the considered system in some sense and, starting from mental models, to generate manifest models (models themselves are also systems). Manifest models serve to support communication about the considered system between different groups of people with different skills.

Information may be obtained by an inquirer by means of direct exposure or by communication. In the case of direct exposure, we must differentiate between situations in which the inquirer regards himself or herself as participating in the development and those in which the inquirer regards himself or herself as merely an observer. In obtaining information, an inquirer will always be influenced by his or her earlier experiences that form the context with reference to which he or she interprets the information. An essential prerequisite for communication about systems are system descrip-
tions. Permanent descriptions must be based on a suitable language containing both natural language and formal elements. All of these must be based on a conceptual framework that is coherent and rich enough for the system description.

In the case of DELTA, the conceptual framework was drawn from earlier experiences with the programming language SIMULA, which was designed as a simulation language for discrete system simulation (cf. Birtwistle et al., 1973). The authors of the Birtwistle et al. (1973) study chose to include the already existing language ALGOL 60 without alteration as a subset of their language, enabling them to concentrate on the development of new concepts. A number of these were to prove trend setting for the further development of programming languages. Of particular interest here is the concept of classes and its related facility for describing and constructing objects that are considered to belong to categories on various levels. Today, this concept is regarded as having initiated object-oriented programming.

With respect to DELTA, this is already evident in the description of what constitutes a component. According to Nygaard's definition, a component is "... characterized by a selected set of associated data items and patterns, and by actions which may involve itself and other components" (Holbæk-Hanssen et al., 1975, p. 15). Characterizations of this sort are termed attributes, and those components sharing the same attributes are said to constitute a SIMULA class. A class does not, then, describe a single object, but rather provides a construction pattern for objects, proceeding from which various instances of objects can be created as needed. The state of a component may then be derived from the values of the quantities and references associated with the component as data items and the associated ongoing action at a given moment of time.

A distinction is made between model time and referent time on the one hand, which refer to the time of events in the model system and the referent system, respectively, and time in the environment of the model generator on the other. The performance of both the referent system's and the model system's actions will occur in some action sequence. The state of a system's components at a given moment of time are depicted by "snapshots" that provide state descriptions.

One instance of a system description is a program running on a computer. Of more general applicability is the use of idealized system generators, in which the developers themselves run through changes in the data items of components in a communication process using suitable aids. For language development purposes, the idealized system generator served, among other things, as a basis for the semantics of the language DELTA, which were not to be confined to the generation of system descriptions in the form of programs running on a computer, but which were nevertheless to be defined unambiguously.
All of what we have discussed so far in this section constitutes an attempt to summarize the initial chapters from Holbæk-Hanssen et al. (1975). In the subsequent chapters of the report, the language concepts of DELTA and their application are described in detail. To go into these in more detail would be going beyond the scope of the present study. To conclude, let us comment briefly on this approach. In our view, it constitutes a particularly elaborate way of dealing with systems in data processing that could well serve as a model for other conceptual approaches. Moreover, Nygaard and his colleagues have continuously extended their efforts to arrive at a better understanding of concepts and methods in data processing (cf., e.g., Nygaard & Håndlykken, 1981) and have themselves gradually taken these beyond the scope of DELTA.

There have been further advances in language development in the tradition of SIMULA, culminating in the language BETA (Kristensen, Madsen, Møller-Pedersen, & Nygaard, 1987), which represents a new level. The DELTA language is a system description language only, allowing description of full concurrency, continuous change, and component interaction. BETA started from the systems concept of DELTA, but is a programming language, drawing upon a large number of contributions to programming research made in the 1970s. An experimental version of BETA has been implemented on Macintosh and Sun, for example. Work continues on designing and implementing a programming environment for BETA as part of the Mjølnner Project (Dahle, Löfgren, Madsen, & Magnusson, 1986), a joint venture in which Norwegian, Swedish, and Danish institutions are involved.

6. SOFTWARE DEVELOPMENT AND ITS METHODOLOGICAL SUPPORT

6.1. Datalogy: Programming Viewed as Theory Building

This section is devoted to consideration of the work of Peter Naur. Naur has been highly influential in shaping computer science education in Denmark, and more particularly at Copenhagen University, where students have been educated along his lines of thought since about 1970. An excellent appraisal of his work and its impact has recently been published (see Sveinsdottir & Frøkjær, 1988). Naur formulated a critique of the formalistic programming methodology in a series of thematically related papers and an introductory textbook (Naur, 1974). Nearly all of these papers are based on experiments or relate to practical experience. In them, Naur opposed the product-oriented software engineering approach that prescribes how software developers are to act without, in his view, properly coming to terms with what programmers actually do.

The experiments related to program development processes on the basis of
predefined problems, these being carried out either by Naur himself or by groups of students under his supervision. Evaluation of the program development processes was done by the diary method: During the ongoing development process, notes on comprehension problems were made together with tentative solutions, program drafts, and whatever other documentation was produced. These were recorded in the actual order in which they occurred along with details of the exact time at which they were produced and how long it took to produce them. In this way, Naur was able to illustrate the kind of activities he and others carried out during program development, pinpointing the difficulties involved and determining the relative importance of the activities (according to the time taken for each).

In his early experiments, Naur took a critical look in particular at top-down design (in the sense of stepwise refinement) and at structured programming. These methodological approaches emphasize the importance of specific abstract program constructs for writing programs and assume (at least in some of the existing literature) a temporal approach to design based on the hierarchy of the product structure. Naur considered published reports on program development processes claiming to have followed this temporal principle to be of dubious value. His own approach to solving a particular problem dealt with in the literature involved quite different work steps (Naur, 1972). He attempted to understand the problem by making use of examples; he correlated individual aspects of potential solutions until a coherent pattern emerged; he worked through a number of potential tentative solutions, elaborating each of them down to a certain level of detail in order to compare their relative advantages and disadvantages; he paid particular attention to input/output formats; to solve subproblems, he wrote connected sequences of statements and checked their correctness by means of examples; finally, he made a highly detailed examination of the presumed efficiency of the desired solution. All of these essential steps in establishing a high quality solution (and a number of others, too) are completely ignored by structured programming. Moreover, these steps are not taken in some predefined order, but rather accompany the developer's gradually growing understanding of the problem he wishes to solve. The quality of the program depends on the care taken in executing and checking these work steps.

Naur (1975) reported on an experiment with a group of students in which several participants each gave an account of the development of the same program using three different case studies. The findings reported in Naur (1972) were substantiated here, and a further discovery was made, namely, that individual programmers adopt quite different approaches to the same problem: Each programmer favored different work steps in a different order; each spent a different amount of time on the various activities; and each derived his or her solution from different trains of thought and individual steps. This questions the value of a predetermined method that attempts to
prescribe which work steps are to be taken and in what order. In his textbook, Naur (1974) therefore rejected the idea of providing a system of rules on how to proceed in programming and, instead, provided a series of useful work techniques and means of representation by using examples to illustrate their application to the student. The aim here was to enable the student to select what he or she considered to be the most appropriate technique for a specific development situation.

Naur (1982) looked specifically at the value of formalization with regard to the correctness of programs. Using as an illustrative basis concrete examples of formal specifications taken from the literature, he demonstrated that formalization alone is not instrumental in ensuring correctness. On the contrary, both formal specifications and those written in normal prose can be correct or incorrect. In the mathematical literature, formalization is not employed as an end in itself, but rather to achieve more precise descriptions. Naur recommended doing the same in programming: Specifications that are to be understood by human beings—and the same applies to proofs of correctness—must, in the final analysis, invariably be embedded in natural language; this is why the use of prose is of particular importance. The overriding aim of comprehensibility must, then, take precedence over the choice of the means of expression. He recommended the use of suitable formalizations, where advantageous, and of readily intelligible examples.

In a further experiment, Naur (1983) examined how errors occur in programming, again using the diary method. Opposing the claim that formal means of representation help avoid errors, he showed that errors in programming most frequently occur at points that are given the least attention by the programmer, presumably because he does not expect to encounter problems there or because he considers them of little relevance for the solution of the problem in hand. This, again, questions the value of formalization for producing correct programs.

Finally, Naur (1985) stressed the fundamental importance of direct intuition for human cognition in all spheres, including the development and understanding of programs. He considered it a serious mistake to underestimate the value of intuition, to regard it as an unreliable factor, or to actually attempt to replace it by a rule-based system of formalisms. Quite the reverse is true: Methods and formalisms are valuable for supporting intuition, otherwise they are of no benefit.

The most pointed presentation of Naur's view of program development is to be found in "Programming as Theory Building" (Naur, 1984), in which the author argued that the essential thing in developing software, whether it be done by individual programmers or in teams, is to build a theory on the particular domain under consideration. The word theory is used here in the sense of the English philosopher, Gilbert Ryle, to denote something like understanding, enabling us to give qualified answers to questions pertaining
to the specific domain of interest, to act intelligently in relevant situations, and to take decisions concerning our actions. Theory is not simply taken to mean developing or having a command of abstract theorems on some complex domain. On the contrary: Theory building, in Ryle's sense, also takes place in everyday situations, for instance, as we gradually become aware of how we wish to furnish our home. Also, one can only be said to have a theory when one is in a position to relate abstract knowledge to situations encountered in real life and to apply it fruitfully.

Applied to the development of programs, this means that the sum total of the communication and work steps that programmers execute in the course of their development situation must lead them to gradually build a theory. This includes their insights into the problem at hand; the gradually emerging solution; and their decisions as to which parts of the problem are to be incorporated into which parts of the program solution, together with their reasons for doing so. In this sense, a theory is invariably linked to human beings. Thus, the claim that programs should be derivable from defining documents proves untenable.

To substantiate his view, Naur referred to experience gained in software projects, in particular concerning the maintenance and further development of existing programs. Attempts to get new programmers, who are unfamiliar with the theory behind a particular program, to further develop this program are bound to fail or result in severe losses in quality. Naur introduced a set of terms that were related to the life cycle of programs and connected with the availability of his team of developers. A program lives as long as there is continuity with respect to team members, making theory building possible. Where there is no such continuity, programs die. In other words, the theory behind them can no longer be reliably deduced. Attempts are sometimes made to revive programs. As a rule, though, this results in the building of a new, modified theory by another team.

This view reflects Naur's rejection of the idea of methods being systems of rules. Naur cited several authors on the theory of science who point out that scientific work cannot be reduced to the application of methods in the sense of systems of rules. There can, therefore, never be one right method for program development in terms of theory building, and there are no formal methods or sequences of work steps that can be definitively defined in advance and imposed upon program development.

This rejection of methods in the traditional sense does not, however, mean that those aspects of programmers' work involving methods are irrelevant. On the contrary, it is desirable that trainee programmers be made fully conversant with a basic repertoire of methods (e.g., model solutions to prototypical problems, description and verification techniques, and structuring principles for complex systems). How this repertoire is used in program development is,
however, a matter that must be left to the discretion of the individual programmer.

This view of programming as theory building also has important implications for the desirable status of programmers, for their training, and for the way in which they see their profession. Naur opposed the production-oriented endeavors of software engineering that require human beings to supply well-defined subproducts; to work as much as possible according to a system of rules; and to be replaceable by other human beings, or even machines. Instead, he draws a picture of the programmer as a professional consultant, comparable to a lawyer, whose main concern is not with supplying products, but with looking after the interests of his clients. A professional training to match this picture would, in Naur's view, have to focus on conducting practical projects under supervision.

The view of programming just outlined may be considered integral in the sense that it takes account of the totality of the tasks to be accomplished. Theory building in Naur's (or Ryle's) sense is closely related to learning processes and emphasizes the value of communication and empirical work. It thus naturally leads to the aspects of project organization in teams (to be discussed in the following section), and it provides the methodological foundations for humanizing technology design where the appropriate values are incorporated into the design process.

6.2. Methodological Work Forms in Systems Development

Methodological Work Forms in Systems Development (MARS) is the name of a Danish research project conducted by scientists and students at Århus University under the supervision of Lars Mathiassen, and at the Roskilde University Center, in collaboration with various industrial enterprises. The MARS project set out, on the one hand, to examine how systems development is carried out in a practical context, and, on the other, to experiment with new work forms in systems development. This empirical research led to the development of a tentative theoretical basis for systems development and corresponding practical methods. The most important findings of the MARS project are presented in Andersen et al., 1986.

In the following section we describe two theoretical approaches: a dialectical approach to systems development and a theory on the activities taking place during systems development. We then look at a number of methodological approaches and techniques with reference to these theories.

Dialectical Approach

To explain the motivation behind his own particular approach, Mathiassen (1987) distinguishes three different approaches to the development and use of
information systems: (a) the hard systems approach, (b) the soft systems approach, and (c) the dialectical approach.

Without going explicitly into the notion of systems, Mathiassen illustrated the hard systems approach by reference to Yourdon's (1982) *Managing the System Life Cycle*. Following Yourdon's approach, the parts of reality relevant for systems development are viewed as stable hierarchies of stable networks of information processes. The following features may be considered typical of this approach:

- One specific system concept is used to approach both development and use of information systems.
- No conceptual distinction is made between humans and machines.
- Attention focuses exclusively on technical information-processing aspects of reality.
- Consideration is given only to rational behavior on the part of the users, intuitive and situation-dependent behavior being neglected.

Yourdon's approach helped in acquiring a detailed and systematic understanding of certain aspects of the use of information that was considered a prerequisite for constructing computer-based systems. It provided a great deal of information of relevance for the construction of technical systems, but failed to look into the alternative options and conflicts confronted in real-life organizations when dealing with information systems.

As an instance of the considerably richer and more general soft systems approach, Mathiassen chose Peter Checkland's (1981) *Systems Thinking, Systems Practice*, which attempts to apply systems thinking to ill-structured problems in human activity systems. A distinction is made here between two kinds of activities: (a) real-world activities involving actors in the problem situation and (b) systems thinking activities involving actors in systematic reflection on the problem situation.

In the first activity, an attempt is made to build the richest possible picture, not of the problem, but of the situation as it is experienced by the actors. In the second activity, this interpretation is then exposed to systems thinking, providing us with an abstract understanding of the problem and of possible solutions.

Checkland suggested that we use different perspectives in reflecting on the problem situation, resulting in different approaches to solving the problem. Finally, our thinking is confronted with the reality of the problem situation, at which point we evaluate and take proper action.

Checkland's approach enabled us to view existing systems from organizational, behavioral, and technical perspectives. It is presented in terms of
general ideas on how to illuminate problem situations. Unlike Yourdon, Checkland actually called for the use of different perspectives to interpret a situation and indicated that we must confront possible solutions with the reality of the situation before applying them.

Mathiassen (1981, 1982) rejected using what he calls "systems approaches" as a conceptual basis for the development and use of computer-based systems. (The notion of system implied here is in keeping with that used by Nygaard and is described in Section 3.) In contrast, Mathiassen favors a dialectical approach that he considers fundamental because it focuses attention on the permanent process of change and sees contradictions as the driving force of change. The systems development approach advocated by Mathiassen is basically inspired by Israel's (1979) book The Language of Dialectics and the Dialectics of Language.

**Contradictions and Contradictory Relations.** Mathiassen proceeded from three ontological assumptions:

- Reality is organized and structured as a whole.
- Reality is in a constant process of change.
- Contradictions are the driving force of change.

Building on these assumptions, he viewed the structure of an organization and the process of its change as complementary, although they can easily be seen as contradictory. Attention should focus here on the process of change.

He believed that contradictory relations within an organization should be emphasized and used as a primary source for understanding and explaining social phenomena. Accordingly, the development and use of information systems is seen as a complicated dialectic, in which the development efforts transform the way information systems are used, and where this process, in turn, transforms the way in which development efforts occur. The production or change of structures is thus largely determined by a series of contradictions:

- contradictions between structures in existing systems;
- contradictions between established traditions and organizational settings on the one hand, and attempts to apply new approaches (e.g., prototyping) on the other; and
- contradictions in terms of social conflicts between the actors involved.

**Processes and Structures.** The concept of process characterizes qualities of a phenomenon that are subject to change in time and space and in terms of development and transformation. The concept of structure, on the other
Figure 4. Basic subtypes of process-structure approaches. The left-hand (right-hand) side indicates that: (a) a process (structure) affects and changes inferior structures (processes), (b) a process (structure)—possibly via other processes (structures)—can affect and change superior structures (processes), and (c) superior structures (processes) can limit and restrain processes (structures). Note. From System Design for Human Development and Productivity: Participation and Beyond (p. 55) by P. Docherty, K. Fuchs-Kittowski, P. Kolm, and L. Mathiassen (Eds.), 1987, Amsterdam: North-Holland. Copyright 1987 by Elsevier Science Publishers B. V. Reprinted by permission.

![Diagram](image)

hand, characterizes qualities of a phenomenon that are seen as fixed and stable.

If we accept this, it follows that we can choose between different types of approaches toward the development of computer-based systems:

- approaches focusing on process qualities (e.g., procedural descriptions of work processes),
- approaches focusing on structural qualities (e.g., descriptions of data and information structures or design of input and output formats), and
- approaches focusing on both process and structural qualities that emphasize the relationship between processes and structures.

Figure 4 shows the two basic subtypes of process-structure approaches. These different perspectives on the same phenomenon with the same qualities enable us to express the complicated dialectics.

Advantages. Unlike the previously described approaches, conflicts, contradictions, and unexpected demands do not surface as problematic exceptions, but are seen as an integral part of systems development. Confusion and lack of understanding are seen as an expression of contradictions between what we know and the situation we face. From this, we hope that new insights may be developed.

Relations identified as being contradictory during development may be seen as an important source of explanations for the problems encountered.
The experience we gain can be turned into knowledge and used to design better conditions and more realistic plans for future developments.

Theory of Activities in Systems Development

The theory of systems development (Andersen et al., 1986) presented here is a theory of the activities taking place during systems development.

System developers perform two types of creative activities: On the one hand, they create a product, the computer-based system (i.e., they create the users' future working process, the software, and so on); on the other, they create a project by determining their own working practices that are to result in the computer-based system (i.e., they create the actual development process). In a sense, we are dealing here with two dimensions: a product-oriented one (related to the system under development) and a process-oriented one (related to the approach to be adopted; see Figure 5).

During systems development, then, both product-oriented and process-oriented activities are performed and both must be considered equally important. These activities are not, of course, carried out separately, but in close conjunction with one another, because both have an influence on the same project. The one kind of activity is termed system development proper or performance; the other is called system development management or project organization. Within these two activities themselves, a further distinction is made between descriptive activities and executive (changing) activities.

The descriptive activities are again subdivided into activities describing things that must be established in the future (design of a product or planning of a project) and activities describing current situations and the consequences of changing them (analysis of the users' working practices, appraisal of technical options, evaluation of a project).

On the performance side, a distinction is made between analysis, design, and realization activities. The double-headed arrows in Figure 5 indicate the interaction between the respective activities. Generation of visions is thus an example of a design activity that interacts with both analysis and realization activities.

A starting point for generating visions might be, for instance, when the idea of installing a computer-based system is first entertained. This idea might originate from the assumption that cooperation between different departments could be made more efficient, this being verifiable by an analysis of the organization. The analysis will, in turn, influence the further generation of visions. The visions generated will lead to determination of the system characteristics and eventually to the system design that constitutes the blueprint for the realization. Experience gained from earlier realization activities will create a need for new or altered design to take account of technical, temporal, financial, and other constraints.

Viewed from this perspective, the technical and functional design is the
most important intermediate product. It does, in fact, constitute the interface between all performance activities. Furthermore, the future users and their management use the results of generating visions as a basis for evaluating whether the design proposals live up to their intentions. This dual application makes it imperative that the results be documented in a form that provides a basis for the technical realization of computer-based systems and, at the same time, enables users to visualize their future working situation.

Analogous to the performance activities, we have, in project organization, the activities of evaluation, planning, and regulation. This view of project organization highlights the fact that those involved in the development process constantly reflect upon their own actions. The activity of planning should include the framing of a project plan indicating how the project is to be conducted and what resources are needed to do so. The aim of evaluation is to keep a constant check on critical external constraints, such as the observance
of deadlines, user participation, user qualifications, and so forth. What is essential here is to gain a proper understanding of the groups interested or affected, including their organizational and technological frames, in order to evaluate and substantiate statements concerning the fulfillment or non-fulfillment of the project plan. Regulation aims at bringing the project and its environment in accordance with the project plan.

Here, the project plan is the most important intermediate product, constituting as it does the interface between evaluation and regulation. This means that precise planning and systematic evaluation, which are frequently neglected in projects conducted along conventional lines, rank at least as high as the regulation of troubles encountered in the course of a project.

The relationship between project organization and performance is characterized by the need for a strong flow of information from the product-oriented development activities to the process-oriented management activities in order to manage the project. Conversely, management of the project and its environment strongly affect the course of the system development proper.

There is a strong analogy, then, between project organization and performance. The importance of the project plan for the project organization corresponds to that of the system design for the performance.

Methods and Techniques

To match these theoretical approaches to systems development, several methods and techniques have been developed, for example, project establishment (Andersen et al., 1986), base lines (Andersen et al., 1986), generation of visions (Kensing, 1987), and project evaluation (Munk-Madsen, 1985). The first three of these are dealt with in more detail next.

Project Establishment. The aim of project establishment is to create, on the part of both users and developers, an awareness and appreciation of the fact that they are working on a joint project. To accomplish this, each side must undertake commitments with respect to the other side and must meet these commitments in the course of the project. The type of commitments may change during the project. From the point of view of the product-oriented activities, what project establishment yields in the way of results is a rough overall draft of the system to be developed. From the point of view of the process-oriented activities, it results in an overall plan for the project performance. In addition, at least one base line is produced during project establishment.

Base Lines. The base lines concept is meant to supersede the phase-oriented approach of classical project models. In the phase-oriented approach, project phases are defined without reference to the specific problem or project in hand and culminate in the generation of a product—a milestone. In
contrast, the base lines concept responds flexibly to the process of development specific to each project.

Base lines coordinate the development process by describing the process on the basis of the existence of specific products. A single base line thus describes a time-independent process state. A base line includes:

- a list of products or intermediate products,
- criteria for checking whether the state has been attained,
- techniques for evaluating the state once it has been attained, and
- fixing a point at which the techniques are activated.

Project planning thus means the definition of base lines. Project planning may be incomplete. For instance, it may be that only one single base line is defined during project establishment; this must, however, contain as its product at least one new base line. In this way the development process can be made flexible.

**Visions Generation.** The notion of visions generation is designed to ensure that greater importance be attached to a frequently neglected part of system design: the joint development of ideas. Visions generation means the development of ideas as to how specific tasks might be accomplished in the future and how computer technology can be used to support them.

The result of this development of ideas then forms the basis for more detailed design activities such as modular design, determination of system architecture, description of manual functions, and so forth.

In order to be able to develop good ideas, the groups involved must be suitably qualified. In this connection, the future users of a system often have to be given appropriate training. The joint development of ideas is particularly important; without it there are invariably obstacles to understanding and translating ideas into technical systems. It is not sufficient simply to consult those affected or to present ready-made concepts in order to set in motion a creative and stimulative design process.

There are hardly any methodological supports for the generation of visions. Potential forms of support have been developed by Kensing (1987). His suggestions address three areas: (a) qualification through future workshops, (b) multiperspective approaches, and (c) negotiations support.

Because the system development process can justifiably be seen as a political process determined by conflicts and cooperation, one possibility is the establishment of future workshops of the sort proposed by Jungk and Müllert (see Müllert & Jungk, 1981). The objective of these future workshops is to make it possible for resource-weak groups to participate in the political
process, to work out suggestions for their ideas on future developments, and to encourage constructive thinking on their part. In system development, future workshops could serve to provide the groups involved with the necessary qualifications to enable them to participate actively in the development process.

By forming work groups that consciously adopt different perspectives, it is possible to obtain a more accurate picture of the organization and of the implications of particular ideas. Such an approach counters the tendency toward a standardization of computer-based systems and increases the users' scope for influencing the design of the system with a view toward optimizing its use.

If systems development is carried out in the manner described, latent conflicts between employers (or managers) and employees are likely to surface. Kensing therefore proposed the following principles for conducting negotiations:

- If management wishes to change work processes, install computer technology, and launch a development project, then workers and trade unions are to be given detailed information about these proposals.

- The trade unions shall discuss the proposals, and both sides shall reach agreement on the project organization.

- Once the project is initiated, one of the first activities is to describe the intended changes and their implications.

- The trade unions discuss the proposals again and reach agreement with the employees on the changes they consider acceptable. This agreement forms the basis for the further project work.

- If management, in the course of the project, wishes to go beyond the agreement, new negotiations are initiated.

Here Kensing assumes that the attitude taken by the trade unions is one that has already progressed beyond an attitude of total opposition to the use of computer-based systems.

6.3. Cooperation Between Developers and Users

Mutual Learning in Participative System Design

The view of the work processes involved in designing and using computer technology that emerged with the collective resource approach was, by its very nature, bound to entail taking issue with traditional participation concepts,
particularly those advocated by the sociotechnical systems school. The role of the users needed redefining, and there were no suitable methods available for actively involving users in the planning, designing, and decision-making processes that the introduction of computer technology entailed.

All the perspectives on the role of users in the design of computer-based work processes that emerged with the collective resource approach share the same basic assumption: that developers and users "should collaborate as a team of different experts" (Bjerknes & Bratteteig, 1987b, p. 325), all of whose expertise is needed. Professional experience and know-how with respect to both computer technology and the work process in which the system is to be used are equally indispensable requirements for a technology and work design that are conducive to improving the quality of work.

System design "should be done with users, neither for nor by them" (Ehn & Kyng, 1987, p. 54). Design for the users (i.e., by the developers) necessarily neglects their specific expertise with respect to their work activities and work organization and generally leads to too much attention being paid to technical, computer-related factors, which in turn results in less demanding, dequalifying jobs.

On the other hand, to put design exclusively in the hands of users without any professional training in systems design is neither an inexpensive nor effective alternative, because nonprofessional developers are unable to gauge the potential and limits of computer technology and are not in a position to ensure suitable embedding of the system into existing work processes.

"Collaboration in system development means performing joint activities, creating joint experiences" (Bjerknes & Bratteteig, 1987b, p. 325). The users are experts in the application area and are familiar with the terms in general use there. Developers are experts in the field of system design and are conversant with the current terminology in that sphere. It is on the establishment of a common basis for communication that the whole development process and, ultimately, the quality of the system depend.

Mutual learning between developers and users is therefore an important part of a development group's work processes. The UTOPIA and Florence projects in particular (see Section 8) tested different concepts for promoting the mutual learning processes essential to design. Both Ehn and Kyng (1987) and Mathiassen (Andersen et al., 1986) call for the conscious initiation of a mutual learning process from the moment a project takes up its work. They suggested that the development group evolve a process of mutual learning in which users become acquainted with the technical potential and limitations of computer technology, and in which developers become conversant with the users' work activities and the know-how required for performing them. This enabled development of a common basis of competence that is of crucial importance, particularly in generating the reality and user models to be transferred to the computer and in designing the system's user interface.
One of several important approaches to promoting learning and the underlying communication processes in computer-based system design (see Bjerknes & Bratteteig, 1987b) is the principle of *learning by doing*, the practical implementation of which uses the specific advantages of computer technology (e.g., by using prototyping).

In work areas where communicative relations are of paramount importance and where invisible, nonformalizable information is crucial for understanding the work processes that are supported by the computer system (e.g., the service sector), Bjerknes and Bratteteig (1987b) suggested encouraging learning processes by means of confrontation:

The communication between different participants, each with interests in the system development process and the computer system, is important. . . . Agreements and differences between the interest groups should appear during the construction of the computer system, not afterwards. . . . In addition to revealing the interests, unveiling different views and objectives in the system development enhances understanding of the application area. By approaching the application area from different points of view, differences between the professions and fields involved will appear in the process. The blindness inherent in the professions could be made visible through confrontations between the different participants.

Creating confrontations is thus a way of learning the premises for collaboration in system development. (pp 325–326)

All the concepts relating to participative system design are based on the same fundamental principles: They must be geared to the users’ work processes; they require methods and techniques for evolving communication and learning processes; and they aim to establish a common basis of knowledge between users and developers.

**Prototyping—A Methodical Approach to Evolutionary System Design**

Recent years have seen, in Scandinavia, the adoption and testing of the technique of prototyping in participative system development for designing computer-based work processes. For this reason, we now take a closer look at the role of prototyping in the system development process.

The use of prototyping in the design of computer-based work processes calls for, and at the same time promotes, user participation. By reference to prototypes, communication between developers and users can be concretized, and the process of learning can be supported. (One possible disadvantage of using prototyping at a very early stage in the development process, though, is that the implementation of a chosen solution may make the users less open to other options.)
We have had considerable interchange with the Scandinavian software developers with regard to using prototyping in participative system development. We therefore discuss prototyping in the following sections along the lines originally proposed by Floyd (1984). Three forms of prototyping are distinguished there, which have been widely adopted in Scandinavia and tested in the context of development projects: exploratory, experimental, and evolutionary prototyping. The boundaries between them are fuzzy, but it is useful to distinguish among them in order to highlight the position occupied by prototyping with respect to the system as a whole in a concrete development situation. One fact that has become apparent is that, when applied systematically, prototyping is useful as a strategy for developing suitable software systems.

**Exploratory Prototyping.** This form of prototyping is designed to provide the different groups involved in system development with the opportunity to discuss the various options open to them. As a rule, the developers have only an imprecise picture of the application area, and the users, for their part, have only a vague idea of how the computer might support their work. This is where exploratory prototyping comes in, attempting as it does to initiate a mutual communication process in the course of which ideas are developed and concretized.

The results of this process are frequently unstructured and without any properly defined relation to the final product. For this reason, one rarely makes further use of prototypes of this sort.

**Experimental Prototyping.** Experimental prototyping is useful for obtaining a better understanding of certain aspects or parts of the final system, and in particular, for producing a more reliable specification. If the prototype is produced in the same development situation as the final system, it may well be used as part of this system.

In the context of participative system development, the most important kinds of experimental prototyping are:

- **Base machine construction**, in which several primitive functions are implemented, and the user is given the opportunity to combine these to produce more complex functions, thus implementing the set of actions and operations required to perform specific work tasks (see Section 8.1).

- **Human–computer interface simulation**, in which the proposed interaction between human beings and the computer is simulated to show the users what the real system will look like, without actual implementation of all the functions (see Sections 8.2 and 8.3).
Evolutionary Prototyping. Evolutionary prototyping is based on the fundamental idea that the organization in which a software system is installed continues to develop, which then results in new requirements. For example, changes in the application context brought about by the installation of the system may themselves give rise to new requirements with respect to the system. For this reason, a dynamic development strategy that views the product as a series of versions is required. In this way each version can be looked on as the "prototype" of the subsequent version. The system development process is seen as a sequence of development cycles, not as a series of development steps to be executed only once in a linear manner. Changed and new requirements can, in each case, be incorporated into the next cycle.

Evolutionary prototyping calls for a high degree of flexibility and willingness to learn and communicate on the part of all those involved. For the developers, it means providing for frequent revision of their work. The users, for their part, must take a positive view of changes and be willing to use and evaluate frequently changing versions that may also influence their work organization.

Assessment. The prototyping process is not controllable in advance, being largely determined by factors such as the willingness of the participants to offer criticism and to exercise self-criticism, and the ability to adopt different perspectives.

Prototyping is useful for assessing, from a user perspective, the extent to which the software system provides adequate support to the work processes. This means considering the functions provided by the system as well as the human–computer interface. It also involves checking whether the system gives appropriate, complete, and consistent consideration to all aspects of the work activities. Prototyping thus helps solve technical problems and promotes learning processes. It also serves to pinpoint any social or organizational problems that may be encountered, thereby enabling them to be given appropriate attention.

The importance attached to prototyping throughout Scandinavia is evident from the wide variety of publications that have appeared there dealing with different application areas (see, e.g., Budde, Kuhlenkamp, Mathiassen, & Züllighoven, 1984).

As a paradigm for software development, prototyping has also been influential in education and training, serving, for example, as a basic approach in courses of study on information systems at Lund University.

7. TOWARD A THEORY OF SYSTEMS DEVELOPMENT

In line with the growing interest in the development of sophisticated and socially oriented computer-based systems, we have looked at several different
efforts to build theories on systems development at various points in the previous sections. Such a theory must contain both descriptive and prescriptive elements. It must concern itself with the development and use of computer-based systems, with the nature and scope of scientific approaches for studying them, and with concepts for combining insights from different scientific perspectives. In this section we examine steps that are currently being taken in this direction in Scandinavia.

The Nature and Scope of Informatics

In a fundamental article of his entitled “Program Development as a Social Activity,” Nygaard (1986) characterized informatics as “the science that has as its domain information processes and related phenomena in artifacts, society and nature” (p. 189). According to Nygaard (1986), “A process is regarded as an information process when the qualities considered are: its substance, the physical matter it transforms, measurable properties of its substance, represented by values, transformations of its substance and thus its measurable properties” (p. 192), and “The most important information processes are, in our present context, program executions and data processing performed jointly by people and computers” (p. 191). Related phenomena include elements such as available tools, program and system development, the learning and work processes taking place, as well as the constraints acting on these.

Nygaard is not alone in advocating an integral approach to informatics. Naur, for instance, also formulated a view of informatics (see Naur, 1974) covering features ranging from the technical aspects of coding to the status of very large information systems in society. Nygaard’s definition of informatics is not, as he himself emphasized, compatible with other definitions that view informatics as a formal discipline, akin, say, to mathematics (see also Naur’s critique in Section 6.1). Instead, Nygaard’s definition is similar to those that apply to sciences such as botany, physics, sociology, and so forth, all relating to selected aspects of specified classes of phenomena. Sciences defined in this way have four aspects:

- Phenomenology—The empirical study of phenomena.
- Analysis—Comprehension and explanation of phenomena in terms of an underlying theory.
- Synthesis, construction, and technology—Knowledge organized for the purpose of interfering with, constructing or generating phenomena.

---

3 Nygaard explains in this article why he prefers to use the term “informatics” as opposed to “computer science.”
• Multiperspective reflection — The consideration and examination of concepts and phenomena at the same time (or alternatingly) from the perspectives of more than one science or from more than one perspective within the same science.

Nygaard, then, called for multiperspective consideration of the domain of informatics and pointed out that the capability of multiperspective reflection is essential for every computer professional. From this, it became clear that perspectivity is a key concept in informatics.

Perspectivity in Scientific and Professional Practice

Nygaard (1986) defined the term perspective as follows: "A person’s perspective is a part of the cognitive universe that may structure her or his cognitive process when relating to situations within some domain" (p. 191). This definition implied the assumption of a cognitive universe that, according to Nygaard, is "the collection of all cognitions of a person."

Bråten, Jahren, and Jansen (1982) gave a simple and plastic definition, betraying the origin of the notion of perspective in geometry: A perspective is a class of related viewpoints, consistent in the angle they adopt on relevant aspects of a specific area of interest (cf. also Floyd, 1987).

What is meant in each case is an interplay of two related insights. One is that our cognition is inherently colored by our respective backgrounds (i.e., our professional or political background, our experience, values, etc.) that influence the aspects that appear relevant to us and that result in different approaches. The other is that, once we understand this, we can make conscious use of perspectivity for elaborating, contrasting, and combining views of the same area of interest from different angles in order to gain deeper insights. This conscious shifting between different perspectives is the basis for multiperspective approaches.

Emphasis of the adoption of different perspectives opposes the idea of a supposedly value free technological development along purely pragmatic lines. In fact, this view itself turns out to be just one perspective among many possible ones, favoring certain developments and excluding others.

In describing and developing complex systems, models are frequently built that are then allocated to a particular system perspective (and that are also differentiated with respect to their level of detail description). In a widely cited article, Bråten (1973) demonstrated the power of models in communication processes.

This model power becomes effective, for instance, in communication processes between two actors, A and B, where A is equipped with a sound model with respect to the domain under consideration, whereas B lacks model resources (A is then termed the “model-strong” actor and B the “model-weak” actor). In this situation, B will inevitably attempt to express himself through
Figure 6. Dialogue models: (a) monological state: A has a model monopoly, ruling out or "swallowing" any other perspective, such as that of B; (b) dialogical state: Different perspectives are brought into a state of complementarity. Note. From Sociocybernetic Paradoxes (p. 199) by F. Geyer and J. van der Zouwen (Eds.), 1986, London: Sage. Copyright 1986 by S. Bråten. Adapted by permission.

A's sounder model. Paradoxically, this means that in situations where B is weaker and anxious to strengthen his or her position, B is actually weakened by attempts to increase his or her influence by internalizing A's model. In other words, B becomes increasingly dependent on A's model. The only alternative open to B is not, as one might expect, to increase communication with A, but to withdraw from the communication situation and develop a different model (i.e., a model from his or her own perspective) in order to return to the communication process as a model-strong actor. In real communication situations, however, this frequently does not occur. Instead, A's model prevails, and a model monopoly is established. In other words, both actors reach a compromise by reference to the same model. This prevents the establishment of a dialogue in the proper sense of the word (see Figure 6).

Bråten applied this thesis to quite different domains, both large and small:
to the futility of efforts to bring about democratization in the absence of an independent workers' model (Bråten, 1973, 1981, 1983); to interaction between two different scientific traditions (Bråten, 1986); and even to direct communication between human individuals.

This diversity is also reflected in the application of the perspective concept by Nygaard and others in the domain of informatics. By way of a supplement to Nygaard's (1986) catalogue of highly diverse applications, we classify the application of the perspective concept in informatics along the following lines:

- the adoption of perspectives in the context of a development project;
- the significance of perspectives in connection with methods in informatics;
- the consideration of the domain of informatics from the perspective of different scientific disciplines; and
- the value-guided adoption of perspectives on technology development with a view to attaining specific sociopolitical goals.

Before going into more detail, it should be noted that no one is capable of giving a complete definition or delineation of his or her own perspective. Essential elements always remain implicit. Nor is there any possibility of our stepping outside our own particular perspective. The important thing about the adoption of perspectives is to make explicit the fact that individuals, or groups of individuals, adopt perspectives and, by combining different perspectives, develop richer insights enabling them to take into account, as much as possible, all perspectives (design on the basis of different and, in some cases, contradictory requirements).

The Adoption of Perspectives in the Context of a Development Project

What is meant here are techniques for identifying and taking into account the angles or viewpoints of those participating in the development project. Kensing (1987), for example, recommended such techniques as an essential element in participative development strategies. The perspectives may be fanned out by allocating them functional roles covering related sets of tasks (cf. Floyd, 1986, 1987; Mehl, Reisin, Schmidt, & Wolf, 1986; Naur, 1974) or by relating them to interest groups, or both of these. The adoption of perspectives may take the form of role playing during the ongoing communication process; or it may take the form of separate models, generated by different groups and not correlated until a later stage, in order to eliminate model monopoly; or it may be reflected in the structure of formal descriptions.

Another way in which perspectives can become effective in the context of a development project is the adoption of a conflict perspective as opposed to
a harmony perspective. This was illustrated in Section 4 as an important distinction between the sociotechnical and the collective resource approaches. The harmony perspective assumes, as a basis for systems development, that a consensus exists between all those affected, and there is no need to take contradictions into account. Any conflicts that may be encountered in the course of the project are ignored as much as possible for being undesirable and immaterial concomitants. By contrast, the conflict perspective views conflicts between the different groups affected as a vital driving force behind systems development, deserving major attention with respect to assessing the soundness of decisions.

Ways in which perspectives are also used in descriptions for adapting existing information systems to user needs—an important problem of direct relevance not in system development but in system use—are demonstrated by Sandström (1987) in her article “How to Improve Pragmatic Quality of Information Systems.”

The Significance of Perspectives in Connection With Methods for Software Development

In his doctoral thesis, Mathiassen (1982) stressed the importance of the perspective as an essential, implicit, and constituent part of methods. He referred to a number of different aspects that cannot be treated exhaustively in our study. These include, for example, the following questions: What assumptions do methods make about the relationship between human beings and machines—about desirable interactions between the social and the technical system? What view on the application of methods is embodied in the respective methods themselves (e.g., open or closed, following a system of rules, or adaptable to the concrete situation)? These aspects are treated in our study in the sections dealing with specific methods: ISAC (Section 5.1), Naur’s critique (Section 6.1), as well as in Section 3.

Explicit consideration is given to perspectives in relation to those issues in informatics that are focused on in Sections 5 and 6:

- Perspectives on the use of computers in organizations include (a) the system perspective, in which the computer is seen as a technical component interacting with other components of the social and technical system in order to attain a particular objective as an overall system; (b) the tool perspective, in which the computer is seen as a work tool for people working mainly on their own; and (c) the communication perspective, in which the computer is used as a means of communication between people working together.

- Perspectives or views of the system development process include viewing it as system development, as software production, as theory
building, and as dialectical process/structure hierarchy. Of course, these are not the only conceivable perspectives.

- Perspectives on the view of cooperation between developers and users include various forms of participative approaches or the absence of such, in particular mutual learning. Another important aspect is the adoption of either a cyclic or a linear view, the cyclic, evolutionary view allowing the provision of technical support for mutual learning.

Consideration of the Domain of Informatics From the Perspective of Different Scientific Disciplines

Of particular relevance here are psychology, sociology, epistemology, and related branches of philosophy and linguistics. Cooperation between different disciplines takes the form of projects, of multiperspective theory building, or results in curricula aimed at multiperspective skill training. There are numerous endeavors in this direction. A few of these are mentioned next by way of example:

- Multiperspective approaches to understanding organizations in which computers are to be embedded (e.g., the Transaction Costs Approach; Ciborra, 1981) or efforts by Bjoern-Andersen and others to arrive at a comprehensive understanding of office work as a basis for possible automation.

- Bràten's proposal to institute a curriculum for socio-informatics that would consider the domain of informatics from the perspective of three disciplines: the informatics perspective, focusing on design and construction elements; the artificial intelligence perspective, dealing with the representation of cognitive elements that can be divorced from human beings; and the humanities perspective, considering semantic associations and their processing by human beings (Bràten, 1984). The idea here is to break the model monopoly of an informatics training focusing exclusively on technical mathematical aspects.

- The tentative cooperation between the systems development group at Århus University (Kyng and colleagues) and scientists working there in the field of linguistics, whose aim is to elucidate the role of communication and language in systems development from the perspective of both of these disciplines.

The Value-Guided Adoption of Perspectives on Technology Development With a View to Attaining Specific Sociopolitical Goals

The basic idea associated with the notion of perspective (according to which the methodological approaches, techniques, tools, and so on, employed in
work and technology design invariably reflect a particular perspective) has had a profound influence on the collective resource research approach. Of particular importance has been the demonstration of the fact that methodological approaches that fail to take account of the objective conflict of interest between labor and capital (conflict perspective), and that consequently reflect an accord of interests (harmony perspective), are unsuitable for carrying out system design from the workers' perspective. "Not to choose is to choose the harmony perspective," stated Nygaard (1987, p. 194). Nygaard (1987) also said that:

Up to 1967, when the Norwegian trade unions started taking an interest in the application of information technology, system development methods in Scandinavia were always presented without any reference to possible conflicts between capital and labour. They were regarded as "objective", "neutral", "professional", "apolitical". . . . We are instead building upon ways of thinking (models) in which one may include both common interests and unresolved conflicts of interests. In our opinion this is necessary, since the objective of actions from the trade unions is a change in the power relations in companies and in their organization. . . . In our opinion language, concepts, models and theories for organization, job content and society are reflecting the interests and ideologies of those who created these languages, concepts, models and theories. (p. 40)

The Emergence of the Research Program "Computer Support for Cooperative Work"

At the time our original study was being prepared, we encountered the phrase "computer support for cooperative work" in our conversations with Bjørn-Andersen in Copenhagen and as a new research topic for the interdisciplinary work of the systems development group at Århus mentioned previously. Since then, computer support for cooperative work has gained tremendous momentum as an international, interdisciplinary effort geared to the development of socially oriented computer-based systems based on sufficiently rich epistemological foundations.

Key centers in Scandinavia promoting efforts in this direction are the research group headed by Nygaard at Oslo University and the systems development group at Århus University.

The research program as a whole is being pursued by an internationally linked network of scientists, including, among others, researchers in California such as Terry Winograd at Stanford and Fernando Flores, an independent entrepreneur at Berkeley (Winograd & Flores, 1986); the Intelligent Systems Laboratory at Xerox PARC, notably Lucy Suchman and
her colleagues (see Suchman, 1987); and the group headed by Don Norman at the University of San Diego. There are also several centers emerging in Europe (e.g., the EuroParc research institute at Cambridge and a number of scientific research groups in Northern Germany, including ourselves).

Although we fully appreciate the importance of the new research paradigm, we are unable to do proper justice to it by way of an afterthought to an almost completed study. We must therefore confine ourselves to sketching some of its key elements:

• finding richer epistemological foundations for design, drawing on such diverse traditions as hermeneutics, dialectics, language philosophy, activity theory, and the theory of self-organizing systems (cf., e.g., Budde, Floyd, Keil-Slawik, & Züllighoven, in press; Ehn, 1988; Winograd & Flores, 1986);

• understanding the nature of cooperative work (Bøgh Andersen et al., 1987; Sørgaard, 1988);

• exploring the role of computer-based artifacts in skilled work (Ehn, 1988); and

• trying to find adequate technical means for realizing innovative systems, for example, on the basis of object-oriented programming.

We expect this effort to expand and gain in importance in the near future.

8. DEVELOPMENT PROJECTS—CASE STUDIES ILLUSTRATING NEW APPROACHES

In the light of the new Scandinavian approaches in computer science reviewed in the previous sections, let us now consider, by way of example, three of the projects that have been conducted in recent years. Two of these were Norwegian projects, the other was a joint Danish-Swedish venture:

• Municipal Town and Country Planning,\(^6\) a project concerned with the development of a computer-based system for use in the town and country planning departments of small to medium-size municipalities.

• Florence, a research and development project aimed at development of computer support for work processes in an Oslo hospital.

---

\(^6\) The project has no official name.
• UTOPIA, a collective resource research project whose objective was to
design computer-based work processes for the Danish and Swedish
newspaper industries.

The Municipal Town and Country Planning project was conducted by the
Norwegian Computing Center in collaboration with the administrative staff
of three different municipalities. Prototyping was employed in the project as
a system development strategy. The aim was to increase efficiency in the
manufacture and use of the system by actively involving the users in the
design of the software embedment. Two versions of the system are already in
use, and the initial results are available.

The Florence development project was conducted by the University of Oslo
in collaboration with a number of hospital wards. The project may be seen as
belonging to the collective resource approach, though, in this case, there was
no direct union participation. The objective of the project was to examine the
specific methodological and conceptual requirements for designing computer-
based work processes in which communication and interaction are considered
crucial factors. Initial findings are available.

In the UTOPIA project, computer and social researchers worked together on
a joint basis with graphics workers. The objective of the project was to
contribute to the development of powerful skill-enhancing tools for graphics
workers, extending existing knowledge on the part of the users, so as to
increase quality of work and products. The project was concluded in 1986
after running for a 4-year period, and initial results are also available.

8.1. Municipal Town and Country Planning—Enhancing User
Skills With the Help of Prototyping

This project, which was concluded in 1986, was concerned with the
development of a computer-based system for use in the town and country
planning departments of small to medium-size municipalities. Norway has
about 400 municipalities of this size, a large percentage of which constitute
the market for a system of the type that will eventually become commercially
available.

The project was supervised by Kari Thoresen and Tom Chr. Pape of the
Norwegian Computing Center in Oslo (Pape & Thoresen, 1987). For the
development of the system, three municipalities were selected: Farmland,
Coastland, and Industryland.

Key Questions

The aim of the project was to examine what sort of system would be
suitable for use in the similarly organized planning departments of different
municipalities, and from this, to derive specific criteria for the development process.

The design objective was to provide a tool for more effective work in the town and country planning departments, whose duties can largely be described as consisting of office work. Here, the automation aspects were to be subordinate to the objective of effectivization.

This involved asking questions about "... where in the work procedures computer tools will be useful and how they should be designed to fit in the structure and complexity of office work ..." (Pape & Thoresen, 1987, p. 299). To answer these questions, an underlying model of the office work was produced which was to form the basis for exploring the possibilities of effectivizing and improving the work of the planning departments. The following problem complexes relating to the choice of the development strategy are of particular relevance to the present study:

- The idea was to develop a "common system" (i.e., one developed largely on the basis of centralized competence, but for use in several planning departments with relatively similar tasks). At the same time, local differences between the municipalities had to be taken into consideration: size, type, and amount of building activity, relations to the political institutions, office traditions, and so forth. The system, then, had to be flexible enough to allow adaptation to local variations, while still ensuring quality and effectivity of work without restricting the users' freedom of action.

- Another essential factor was the need to economize with the available development resources. Design of the development process was to be such as to ensure establishment of the necessary skills in each municipality to enable the organizational and technical changes to be handled by the local staff without the need for support from outside professional developers. The important thing here was to qualify the users so as to enable them to use their skills effectively as a development resource; most municipalities in this category do not have the necessary financial resources to employ development experts on a permanent basis. The issue of user qualification was therefore one of central importance in the system development.

**System Development Strategy**

The specific issues and problems just outlined call for a particularly flexible system development strategy encouraging the active participation of the users. The project was approached from two sides. The first was a successive development of pilot versions of the system in the three municipalities (i.e., Farmland, Coastland, and Industryland), using evolutionary prototyping.
The second was a national survey to identify the spectrum of variations relevant to the design and use of a common system (see Figure 7).

By prototyping, Pape and Thoresen (1987) mean:

... the process of developing a prototype, which is a subsystem of the production system to be used for testing and evaluation in a real work setting. Prototyping consists of four steps ... functional selection, construction, evaluation and further use. These steps are repeated until the quality is good enough. The last prototype is then the pilot version of the production system. In this sense a pilot version is intended to have a longer useful life than a prototype. ... It is better designed but is still not complete, and will be gradually extended into a full operational production system. (pp. 303–304)
This view is directly related to the work of Floyd (1984).

The Development Process

The analysis was carried out in Farmland, enabling construction of the first prototype. This prototype was then tested in production and evaluated. A representative from Coastland also participated in the evaluation process to learn about the system and to identify the areas of system design where changes were needed for Coastland.

The experience gained from Farmland (relating to training, organization, etc.) was used for development and installation of a second prototype in Coastland. The process of evaluation was then repeated, this time extended to include aspects of the changed requirements.

These two prototypes provided a common core, with a set of options to account for local differences. A third and last prototype was then constructed, incorporating this core and the specific features of Industryland. This prototype, together with the experience gained in the three planning departments, finally led to construction of the pilot system.

Results

Replacing traditional system development methods by prototyping proved beneficial in determining specific local requirements. It was possible to concretize these and to incorporate them directly into the system development process. This type of system development turned out to be time consuming, though, because the developers were frequently reluctant to abandon well-known models, of which they had a good grasp, and to develop new techniques and approaches.

Problems were also encountered in project planning, because the iterative nature of the development process selected made it difficult to determine in advance when which prototypes incorporating which functional and performance features would be available. And yet here lay the advantages of the development model with respect to the adaptability of the system. By not proceeding from partially predefined results, it was possible to respond flexibly to requirements and wishes that arose in the course of the development process.

The overall development strategy was more strongly process oriented than product oriented. Consequently, the products emerged during the use and evaluation processes, allowing them to be flexibly adapted to these processes.

The somewhat anarchistic approach underlying this strategy initially entailed difficulties for the users, too. For most of them, the idea of the dynamic changeability of system versions was unwonted and difficult to follow, which meant that they initially found it hard to intervene in the changing process. Bringing their influence to bear was quite a different matter, though, once they had acquired greater knowledge through their
practical experience in handling the prototype. Then, they were frequently the ones to assume the role of “changer” in their work organization. It is interesting to note that differences in the levels of the newly acquired qualifications led to social conflicts: A new development elite emerged, bringing with it shifts in internal skill and power structures.

Some of the staff used the skills acquired during the system development process to leave the organization and to look for a better paying job more in keeping with their newly attained qualifications. This represented a serious setback and threatened the success of the project because, after all, one important aim of the project was to establish in the respective planning departments a skill basis that would enable the local staff to make changes and adjustments themselves.

The choice of the system development strategy and the selection of three different municipal town and country planning departments for the development of a pilot version resulted in a flexible system version whose advantages with respect to increasing work effectivity have yet to be evaluated. Previous experience with other systems, especially in the government and municipal sectors, have shown, however, that the choice of a particularly well-suited office for developing a standard computer-based tool that can be used in different local settings results in inflexible systems that are not amenable to effective use in the respective local context.

8.2. Florence—Supporting Communicative and Cooperative Work Processes

The research and development project Florence represents an attempt to develop computer support for the work activities of nurses in a hospital ward (Bjerkenes & Bratteteig, 1987a). The Florence project is part of the Nordic research program SYDPOL (Bjerkenes & Bratteteig, 1987b). Begun in 1984, it was conducted by Gro Bjerkenes, Tone Bratteteig, and Jens Kaasbøll of the Institute of Informatics at Oslo University in collaboration with the nursing staff in the ward of a hospital near Oslo. One of the nurses in the ward was chosen as the staff representative and was relieved of normal duties in order to enable her to participate in the project. The project was due to run until the end of 1987; an initial system version has been in use since late 1986.

Key Questions

The objective of the project was to facilitate and to efficiently structure the daily work of the nurses (the nursing staff are all women) by installation of a computer system. Work in the hospital ward consisted largely of nursing activities that, by their very nature, are unsuitable for automation. The work procedures capable of computer support are not of a tangible, but of an
informational nature, though only a limited number of the communicative and interactive relations conveying this information can be formalized. A substantial portion of the information-processing procedures include the collecting and registering of information, followed by suitably processing and distributing it. In other words, the primary concern is not to generate something new from existing information, but to manage and distribute the existing and rapidly changing information in a suitable manner. Of crucial importance for the daily work is the cooperation between the nurses themselves (between the shifts) and with other groups (e.g., doctors, patients, and nurses in other wards).

The system development's aim was to find computer support capable of being embedded in the existing work processes in such a way that complete control of the information would remain in the hands of the nurses. Management and distribution of information were to be facilitated, and care was to be taken not to destroy existing organizational structures of communication and interaction.

**Methodological Approach**

This project, too, endeavored to make use of existing know-how on the part of the nurses and to maintain their skill levels. Unlike the industrial production sector, where the objects being processed are of a concrete nature, here the tool perspective proved unsatisfactory for considering the work processes involved. After preliminary analyses of the work procedures, the developers decided to carry out system design from an application perspective. Accordingly, they focused chiefly on the nurses' communication processes employed during their work and on those utilized among all the groups involved in the development process.

The application perspective, which involved studying the communicative relations and roles as well as the information flows and processing procedures, led to the choice of a specific approach to system development. According to this approach, work processes are seen as communication processes in the course of which work results are produced. In particular, the system description process itself is seen as a communication process during which a common result is obtained. Here, a wide variety of means and techniques for description may be used, provided they are suitable for supporting those involved in their communicative efforts to attain a common understanding of the nature of the computer system and the work processes.

The developers used formal means for describing the products of system design whenever they wished to specify work results for their own purposes, in order to detect deficiencies requiring clarification with the nurses, and when they set out to implement the results obtained (i.e., actually constructing the system). Here, they used conventional methods and means of description (e.g., ISAC, Yourdon's Structured Analysis/Structured Design,
or Jackson System Development) that are designed to facilitate formal
description and specification of the work procedures to be automated.

The Development Process

The initial project phases were concerned with establishing a suitable
design approach for a computer system that would meet the nurses' require-
ments. Analysis of the work processes was done by both nurses and developers
in close cooperation. Attention focused here on a detailed study of the nurses'
duties, and on experiments with different system development methods. To
enable the nurses to participate in the processes of requirements analysis and
system design, weekend seminars were arranged. Using a variety of methods
(e.g., training on personal computers, role playing, wall displays, etc.), the
nurses were imparted the necessary know-how to enable them to work with
computers. At the same time, communication processes were initiated via
system descriptions using different means of description (e.g., prose texts,
graphics, and formal and semiformal means of representation).

The developers did not, though, confine themselves to observing work in
the hospital ward in order to ascertain which tasks the nurses were required to
perform. They got the nurses themselves to brief them so as to help them
understand the most important aspects of daily work in the hospital. The job
of the ward nurse who was exempted from normal duties was to initiate
communication processes between the nurses themselves regarding the vari-
ous perspectives and design options for the computer system and the work
processes.

One of the things the nurses had hoped to achieve by the introduction of
computer support was to avoid having to enter the same information on
between three and seven different forms. Another requirement was that the
computer screen should give as exact as possible representation of the hospital
ward. Here, the developers ran into technical difficulties when trying to
develop a suitable prototype, because there were no screens available with the
required size and high enough resolution. It proved impossible to fit some of
the forms on a computer screen page; though, for the nurses, it was essential
to have the entire form in view at the same time. The constraints imposed by
the available technology thus made it necessary to implement a nonoptimal
solution.

Results

As a general principle, the choice of a suitable perspective for viewing the
design process must take its cue from the work processes to be designed;
dependent on the choice of perspective are the methodological approach and
the selection of the means of description to be used. For designing the work
processes of a hospital ward, where the emphasis is on management and
distribution of information, it proved most suitable to focus on communica-
tion during work. Particularly important in this respect is the fact that it proved possible, by initiating mutual learning processes based on communication, to establish a common basis of knowledge and a common view of the system. The basis for communication on system design is established by the communication processes taking place during the system description process itself. Here, system descriptions serve as means of communication.

“Our experiences have been, that system description processes differ according to differences in their objectives” (Bjerknes & Bratteteig, 1987b, p. 321). The conscious distinction between the construction of a system description (product) and the use of the system description as a means of communication (process) leads, according to Bjerknes and Bratteteig (1987b), to more specific use of the means of description and better understanding of the system construction. One of the tasks undertaken by the developers, in the course of developing a further system version for another hospital, was the attempt to overcome the technical constraints by using suitable fifth-generation technology.

8.3. UTOPIA—Design of Computer-Based Systems for Enhancing Work Quality and User Skills

UTOPIA is a research and development project employing a trade-union-oriented work and technology design approach for developing a computer-based text and image processing system for the Scandinavian graphics industries. UTOPIA is an acronym in the Scandinavian languages for “training, technology, and products from the quality of work perspective.” The project was launched in 1981 by the Nordic Graphics Workers’ Union and conducted on a joint basis by researchers and skilled graphics workers. The overall objective of the project was to explore the prospects for interdisciplinary cooperation on work and technology design for computer applications. The basic requirements with respect to design were that it should help:

- improve the quality of work and products,
- realize democracy at work, and
- enhance user skills.

One important aim here was to look into the possibility of a technology design that did not necessarily involve cuts in jobs and dequalification. Attention focused, therefore, not only on technology development itself, but also on the issue of user skills. The project set out to determine the basic technical and social prerequisites for trade-union-oriented work and technology design and, at the same time, to identify existing obstacles to, and
limitations on, its realization. More specifically, it was concerned with the work processes of page make up and image processing in integrated, computer-based newspaper production.

Graphics workers and computer and social researchers worked together on the joint Swedish–Danish UTOPIA project that was carried out at the Swedish Center for Working Life and the Royal Institute of Technology, both in Stockholm, and at Århus University. Besides participating directly in the project, the Scandinavian graphics workers’ unions followed and supported the project through a group consisting of representatives from Denmark, Finland, Norway, and Sweden.

The project’s research contacts consisted mainly of other graphics industry research projects and groups doing basic research in computer graphics in Sweden, Denmark, and Norway. At various stages, the project cooperated with the supplier Liber/TIPS (a Swedish company), the Swedish newspaper Aftonbladet, and the Danish newspaper Information.

The project began in 1981 and ran for a 4-year period. An average of 15 people participated in the work at any one time, though most did so on a part-time basis; total input amounted to 15 people-years. The total cost of the project was approximately \$400,000, which was paid for by the Swedish Center for Working Life, the Swedish Board for Technological Development, the Swedish Royal Institute of Technology, the Danish National Agency of Technology, Århus University, and the Nordic Graphics Workers’ Union.

Project work included production of a complete requirement specification for an integrated computer-based system for text, image, and page make up, elements of which were incorporated into the commercial Text and Image Processing System (TIPS) of the Swedish state owned concern Liber, but not technically realized.

UTOPIA may be seen as belonging to the tradition of the collective resource research projects carried out in Scandinavia in the 1970s. It initiated a new project generation that went further than earlier projects in examining issues relating to new alternatives in technology development, work practices, and professional training.

**Design Methods and Techniques**

The design methods behind the UTOPIA project may be characterized by the tool perspective, as opposed to the system perspective (cf. Bødker, Ehn, Kammersgaard, Kyng, & Sundblad, 1987; Kammersgaard, 1988), and the principles underlying the development process paraphrased by the key words *mutual learning and design by doing* (Bødker et al., 1987).

**The Tool Perspective.** The tool perspective is based on the idea that a skilled, experienced worker possesses both explicit and tacit competence
(cf. Bødker, 1986; Ehn & Kyng, 1985a, 1987). Through professional experience, often rooted in a specific tradition, the worker has acquired background knowledge that is essential for doing the job and that cannot be formalized.

Applied to newspaper production, this knowledge might be the ability to arrange articles, advertisements, pictures, and so on, in such a way as to produce a well-balanced, aesthetically pleasing page layout. For this purpose, the raw materials (paper, graphics, and texts) are transformed with the help of tools (ruler and scissors) into a refined product (newspaper page).

From the tool perspective, then, computer support must be designed in such a way as to enable the user to continue employing one's explicit and tacit skills. This becomes impossible if control over the work process is transferred to the computer system (i.e., if the process is automated). That would inevitably lead to a loss in product quality and a dequalification of the skilled worker, because there would no longer be scope for full use of his or her skills.

Adopting a tool perspective therefore implies an attempt to preserve the work processes and skills of the experienced worker. Computer-based tools must allow the worker to develop and to employ a repertoire of work actions and operations. In addition, they must not attract the worker's focal awareness, in other words, distract him or her from the objective of the work activity. The computer system (i.e., the hardware and software together) must represent and actually extend the scope of traditional tools. It must provide a “tool kit” of powerful tools that are suitable for enhancing the worker's skills and increasing his or her qualifications. The computer-based tools should, like traditional ones, be capable of individual use according to the respective worker's needs.

Design of computer technology from the tool perspective is thus governed by the following principles:

- Good computer-based tools must allow the users to develop and to apply their own repertoire of actions and operations in their work.
- A computer-based tool should not attract the focal awareness of the user in regular use situations.
- Good computer-based tools should rely on and help develop the tacit and explicit competence of the user.

Scope of the Tool Perspective. The tool perspective is associated with a development view based on real, tangible tools (drill, lathe, etc.), according to which the tool should support the worker by providing a meaningful extension of his or her skills. This view only holds, however, for support in
generating or refining a product (e.g., a newspaper), though this product need not necessarily be of a purely tangible nature. The approach can also be applied to office work, where in some cases only information is processed, because here, too, a refinement process based on individual competence takes place. It is, however, unsuitable for developing a network or an operating system.

**Mutual Learning.** Assuming, as we do when adopting the tool perspective, that the user is the person qualified to deal with a particular development problem, it follows that the process of developing a computer-based tool must involve both computer professionals and skilled workers from the relevant domain. The users contribute their skills and competence as a basis for analysis, and, in return, they learn something about the technical possibilities of computers. In so doing, they should learn to use their imaginations in exploring the various technical options.

The computer professionals learn about the work of the profession in question and make suitable proposals. Whereas a particular tool might appear to be a simple matter from a user point of view, its technical realization might prove extremely complicated or even impossible. To take a real example, let us look at the problem of representation encountered during development of computer support for page make up. From the tool perspective, the requirement was that the computer representation should correspond as closely as possible to the original page. This was not feasible, however, because there were no screens available of the required size and quality or of the necessary resolution. The mutual learning process ensuing from this conflict should lead to finding the best possible solution. This example also serves to demonstrate the necessity of taking hardware factors into account from the very beginning and not postponing this until after the analysis has been completed.

**Methodological Support**

**Use Model.** To cope with the discrepancy between user wishes and technical feasibility, the UTOPIA project developed a so-called use model (see Figure 8) based on the traditional scope of the profession and combining user skills with the technical realization on two different levels.

On the first level, the user's skills and actions are described along with the material and tools used. Applied to newspaper page make up, these include elements such as pasting, shifting, drawing lines, justifying; and paper, pictures, sets of type, photographs, and so forth.

On the second level, a description is given of how the first level can be adapted to the available technology. In this case, the material was represented by icons, the actions by operations such as shift, erase, copy, and so forth. In
addition, new operations were developed that had not been feasible manually (e.g., automatic centering, framing, and alignment).

Representation of the processed product "newspaper page" was realized by means of "lenses," allowing selected parts of the page to be displayed and appropriately enlarged or reduced to ensure that:

- the body text is easy to read for precise allocation;
• the material is shown in natural size to ensure optical and aesthetic balance;

• the whole page can be seen on the display screen for rough allocation;

• a miniature picture of the page can be shown for guidance, without its taking up too much space.

Mock-Up Simulations. To distinguish a good tool from a bad one, the user must experiment with the tool. To make this possible, various mock ups were constructed using paper, matchboxes, and plywood; these were supposed to represent a high resolution display screen, a mouse, a tablet, and so on (see Figure 9).

The graphics worker now does the page make up step by step. For each step, the computer professional draws the corresponding screen image on paper. This series of steps illustrates aspects such as:

• What material is to be shown and how is it to be represented?

• What other information should be displayed?

• Is a menu required, and if so, where is it to be located?
Various technical options may also be discussed:

- Is a tablet with a mouse or stylus needed?
- How many buttons does the mouse need, and where are they to be placed?

Experience from the UTOPIA project has shown that this is a very good way to get started. One is not restricted to specific (available) material; the user is able to take an active part in the design process; and, last but not least, the method is very cheap. One disadvantage of this method is that drawing pictures on paper is very time consuming. For this reason, it was later decided to switch over to an improved simulation using slide projectors.

**Prototyping and Workstations.** There were other important aspects that proved impossible to explore using mock ups: What is the connection between the movements of the pointing device and the movements of the cursor? How is material moved about on the display screen?

Tentative solutions to these and other similar problems were programmed as prototypes on a computer workstation and evaluated in collaboration with the graphics workers. The prototypes served experimental purposes only and were certainly not supposed to constitute early versions of the final system.

The graphic workers participated in the design of the experiments and of course, they were the ones to try out the prototypes. *This way they could try out the possibilities of developing and using the repertoire of actions and operations. These experiences resulted in changes of the prototypes, and also in a better understanding of what are the possibilities, i.e. alternatives, . . . . and limitations . . . . of computers.* (Bödker, 1986, p. 14)

Compared with earlier attempts using mock ups, this approach provided more realistic contact with the medium computer. The graphics workers were able to articulate their demands in a concrete way by using the prototype and working with mock ups. However, to keep programming effort down to a reasonable level, a suitable prototyping environment is considered essential.

**The Organizational Tool Kit.** As a supplement to the simulations, a method for working out concrete descriptions of the work organization was developed. Starting from an analysis of the activities involved in producing a newspaper, all the functions and materials were symbolized with icons that could be fixed to a wall and moved about at will (see Figure 10). A new icon was added to the organization diagram on the wall whenever it was felt that enough was known about a particular work process and its connections to
other work processes. The diagram itself was discussed with the groups affected and was modified when necessary.

The Project Itself

In this section that draws largely on Bødker, Ehn, Romberger, and Sjögren (1985), we go into the background of the project and how it progressed. We do so in order to illustrate the possibilities opened up by constructive cooperation among research and technology institutions, industrial enterprises, and trade unions, even where each of these partners continues to safeguard their own respective interests. This also enables us to highlight the special features of the Scandinavian situation that could well serve as a model for developments elsewhere.

The first project phase was devoted to mapping out technology, work practices, and training in the graphics industries, as well as the prerequisites for developing alternatives. Various scenarios were drawn up (e.g., the graphics workshop, the information society, the newspaper production plant, the quality press, image processing in 1990, and the automated paradise). The work resulted in suggestions for the development of either of two production situations—prepress production of newspapers and the graphics workshop. Visits were made to newspaper production plants, technical exhibitions, suppliers, and research laboratories in Scandinavia and abroad to collect information.

A major aspect in the first phase of the project was the mutual learning
process, in which the active participants—graphics workers and computer and social researchers—established a common “knowledge platform” for the future work. Many difficulties had to be overcome in this phase. The graphics workers (who were used to achieving swift, concrete results in their daily work) found that the work progressed too slowly and lacked the necessary concretion. It also became apparent that the costs of developing technical alternatives had been underestimated. Moreover, it was difficult to get the Swedish and Danish authorities to support the project financially.

After about 18 months, the project succeeded in securing the cooperation of the Swedish state-owned printing concern, Liber. Liber wanted to examine the possibilities for cooperation around the company’s development project, TIPS, that was concerned with developing an integrated computer-based system for text, image, and full-page make-up for newspaper production. In this project, Liber System Ltd. was responsible for project management, marketing, and training; the Swedish company IMTEC was responsible for the development of the graphics workstation and for image processing; and the Finnish company Typlan was responsible for text processing and page make-up.

Those responsible for the project felt that the reason for Liber’s interest in cooperation was that it had to acquire a substantial share of the domestic market in order to be successful in marketing. This would require paying attention to the demands for technology and training posed by the Scandinavian graphics workers’ union. In the UTOPIA project, whose aim it was to formulate such requirements and attempt to realize them technically, Liber/TIPS found a natural discussion partner. Consequently, the cooperation agreement that was signed in 1982 provided that the TIPS project should exploit the competence of the UTOPIA project (work organization, quality of work, training, human–machine interaction, graphics skills, etc.) in its development work. In this way the UTOPIA project obtained the opportunity to try out many of its ideas. The cooperation would also offer valuable experience in exercising influence over a large technical development project of approximately $10 million. The agreement did not, however, force Liber/TIPS to follow the requirements of the UTOPIA project, nor did it interfere with the UTOPIA project’s right to freely cultivate and inform about its viewpoints and requirements that the TIPS system might not fulfill.

During the next year, the UTOPIA project concentrated on requirement specifications. This called for the development of working practices that would enable the researchers and the graphics workers to formulate the requirements together. For this purpose, the project established a technology laboratory with development tools (mainly those for mock-ups described earlier) to simulate different kinds of page make-up, image processing, and the surrounding organization. This made it possible for the graphics workers to develop requirements and wishes on a concrete level by actually carrying
out the page make up and image processing on simulation equipment. The researchers' contribution was to point out possibilities and limitations to similar equipment and to systematize the experience in requirement specifications.

The first version of UTOPIA's requirements specification, which was continually discussed with TIPS, was published in late 1983. The requirements were meant to be generally applicable in negotiations by the unions when introducing new systems for newspaper production. They were meant, then, for all suppliers, not just for Liber/TIPS. The requirement specification has attracted a lot of attention in graphics trade journals, at seminars and conferences on newspaper production technology, and in connection with collective bargaining and local negotiations.

The next step was, while still working in the technology laboratory, to direct the project resources toward professional training. Of the more than 20 reports produced in the course of the project, the majority were written to be applied in professional training of graphics workers. From 1984 onward, a large portion of the material was also used in the graphics industries department of the Royal Institute of Technology in Stockholm. Valuable experience was gained by using the development tools of the technology laboratory in teaching situations.

At a conference in May 1984, the UTOPIA project and the Nordic Graphics Workers' Union presented the results of their work to graphics workers, journalists, and administrative staff and their unions. Since then, both teaching materials and development tools have been at the graphics trade's disposal.

The cooperation with Liber/TIPS also included a second phase—evaluation of the TIPS system and development of work organization in connection with the first pilot installation at the Swedish newspaper Aftonbladet. During this phase, attention should also have been given to the way in which the newspaper publisher used the new technology, how well it worked, and how the work organization was changed by its use. Unfortunately, the management showed little interest in a well-defined experiment on work organization in connection with the pilot installation. Their attitude was: "Let's get the technology, and we'll see what happens" (Bødker et al., 1985, p. 4). The ideas of the UTOPIA project for active participation had to be abandoned. These conflicts contributed to the near failure of this part of the project.

The overall importance of the UTOPIA project can be summarized as follows:

- It has helped pinpoint and concretize problems specific to the attempt to make use of new technological development lines for designing work with a view toward improving work quality, maintaining and enhancing skill levels, and achieving more democracy at work.
• The methods and concepts elaborated during the project have encouraged and promoted research and development efforts in the field of software development for computer-based working processes.

• Along with the project results, it has provided criteria for assessing the quality of human–computer interaction in the context of everyday industrial working processes. These are relevant not only for the Scandinavian graphics industries, but are of more general significance (see Ehn, 1988).

9. CONCLUSIONS

The approaches and projects presented in our study fail to reflect the whole breadth of Scandinavian research and development work in software engineering. We were unable to consider, for example, important developments in the areas of industrial design and manufacturing. Nor was it possible, within the scope of the present study, to address the growing trend toward networks and knowledge-based systems in Scandinavian research. The study does, nonetheless, present the views and ideas of a representative cross section of advocates of the Scandinavian Approach who openly profess their commitment to socially oriented technology design.

Recent years have seen a discussion of the merits of a technology design geared toward improving the quality of work and products, toward maintaining and enhancing existing skills, and toward establishing industrial democracy (the key phrase here being "new Scandinavian model"; see Ehn & Kyng, 1987). The remarkable thing about this model is, in our view, the fact that it reflects not only the perspective of the workers and their organizations, but also the specific market advantages for Scandinavian high tech industry.

The idea may be outlined as follows:

• technology which supports good working conditions and good use quality products should be developed
• the trade unions should play an active role in formulating requirements to this technology, since they are best fitted to capture and draw upon the employees' knowledge and experience of work and working environment
• this helps developing technology which satisfies the demands of the employees
• this is a unique opportunity and resource which the Scandinavian high tech industry should utilize
• and the governments should foster and support such activities with national and Scandinavian research and development programmes
• furthermore, the traditionally calm labour markets make many sectors of Scandinavian industry good "test sites" for these technological and organizational alternatives
• and give opportunities for domestic markets . . . for at least initial production of this kind of technology, especially if the demands for it are supported by the trade unions and by government requirements
• in the long run this kind of technology which supports quality of work and products will give Scandinavian industry opportunities in foreign markets. (Ehn & Kyng, 1987, p. 48)

What can we learn from the Scandinavians? It goes without saying that sociocultural conditions cannot be transferred from one country to another. None of the historical developments, the sociopolitical distribution of power, and the design culture, which has emerged against this background, can simply be adopted elsewhere any more than can the consequent sociopolitical consensus with respect to a technology policy that is geared to the humanization of work and the democratization of industrial relations.

One thing we can learn from the Scandinavians is that new computer technology opens up new prospects for workers and their organizations. By actively exercising their influence on the planning, designing, and control of technology and work, they can make progress toward the humanization of work and the democratization of labor relations. A factor favoring the exertion of such influence is the dependency of capital on exploiting workers' knowledge and experience with respect to the work processes that are to be automated and redesigned. Another factor in their favor is the productivity benefits, both for the economy and for society as a whole, brought about by socially oriented design and use of technology.

One basic prerequisite for developing a technology of this sort is the evolution of a value-guided scientific and research tradition, providing the theoretical and methodological basis for a technology design that is geared toward humanization and democratization. Another is the need for the trade unions to frame their own policies and strategies in science and technology and base them on their own independent research and development projects.

A further lesson we can no doubt learn from the experience of the Scandinavians is that cooperation between trade unions and universities can most certainly be in the interest of both sides, helping both parties to push ahead with scientific and technological developments and, at the same time, to solve structural problems in the economic and social sectors.

What are the implications for computer science in other countries? A number of the concepts and models looked at in this study could well be adopted elsewhere, at least in part. This is true of the theoretical and methodological approaches, the system development models, and the concepts
for encouraging user participation with a view toward humanizing technology and work design.

What aspects are considered suitable for adoption from the trade union point of view must, in the final analysis, be left to the unions themselves to decide. We therefore confine ourselves to outlining some of the experiences gained and conclusions drawn from 15 years of practical trade union research and development work in the computer field, experiences that have been widely discussed in Scandinavia.

To achieve the goal of active codetermination by the trade unions, project planning must provide for the establishment of the organizational structures needed to guarantee and to foster communication between the groups involved. It is not enough simply to rely on legal provisions during system development.

Basic requirements that the trade unions must fulfill in order to ensure successful interest-governed and action-oriented participation of workers in system development are:

- conducting their own research and development activities in order to acquire and disseminate knowledge;
- evolving, independently of the current development situation, a parallel process of learning for planning and carrying out system development and using new technologies;
- developing suitable negotiation models and positions in order to influence the decision-making process;
- carrying out qualification measures with respect to the planning, design, and control of computer-based systems; and
- providing external resources (e.g., specially engaged consultants to support workers in concrete design activities).

In practice, considerable difficulties are encountered by the unions in their efforts to realize their participation demands:

- Financial and personnel resources are limited, especially at the local level.
- Decisions frequently have to be prepared on short notice.
- Many workers lack experience in using formal means of description and in carrying out complex planning and organizational design activities.
Many workers have difficulty externalizing implicit knowledge.

Most workers lack insight into the real aims of management.

For these reasons it is not, generally speaking, possible to secure participation in all areas of the development process. Such participation is frequently limited to those areas that the unions see as impinging particularly strongly on workers' interests, for example:

- issues affecting jobs,
- enhancement of workers' qualifications,
- design of work processes and work organization, and
- design of the work environment.

On the whole the trade unions are, particularly at the local and company levels, reliant on external financial and technical support in developing their own design approaches with respect to new computer technologies and in activating the workers accordingly.

One thing is certain — The methodological approaches to socially oriented technology design that have been successfully applied in Scandinavia can only be effective in other countries if there, too, a broadly based process of communication and mutual learning is initiated in all sectors of society.

The success of this venture depends, no doubt, on the sociopolitical climate. A factor of crucial importance here is the way in which antagonisms, conflicts, and disagreements are dealt with in general. The ability to admit the existence of contradictions without getting bogged down in them, and to see them as a dynamic force stimulating change in the scientific and social sectors, means — and this is, in our view, the essence of the Scandinavian Approach — recognizing and taking up the technological challenge that they imply.

To this extent, we see our study of Scandinavian approaches as a contribution toward developing a theoretical and methodological basis for socially oriented technology design outside Scandinavia.

Acknowledgments. While working on the present study, we undertook two field trips to Scandinavia. The first took us to Århus, Roskilde, and Copenhagen in Denmark; to Oslo in Norway; and to Lund in Sweden. The second trip was to Sweden, where we visited various institutions in Stockholm. On both of these occasions, we were received with a degree of kindness, helpfulness, and hospitality that exceeded the professional nature of our visit. We take this opportunity to express our deep gratitude for the reception accorded to us.

We are especially indebted to all those with whom we held discussions during our two visits, namely, Morten Kyng and Lars Mathiassen at Århus University; Keld
Bødker and Finn Kensing at the Roskilde University Center; Niels Bjørn-Andersen at the Copenhagen School of Economics and Business Administration; Stein Bråten, Kristen Nygaard, Gro Bjerknes, and Tone Bratteteig at Oslo University; Tom Chr. Pape and Kari Thoresen at the Norwegian Computing Center; Hans-Eric Nissen, Agnete Olerup, Per Flensborg, and their colleagues at Lund University; Mats Lundeberg at the Stockholm Business School; Peter Docherty and colleagues at the Swedish Work Environment Fund; and Bernd Hofmayer at the Swedish Center for Working Life.

In addition to those mentioned here, a great many other colleagues, with whom we are linked by long-standing professional and personal ties, have, through their work, made a significant contribution to this study. Our sincere thanks go to them, too.

Our special thanks are due to Philip Bacon for the great care taken by him in translating the original German text into English.

**Support.** The study was funded by the Government of North Rhine-Westphalia under their program, *Mensch und Technik: Sozialverträgliche Technikgestaltung* (Man and Technology: Socially Oriented Technology Design).

---

**REFERENCES**


Bødker, S. (1986). *User interface design*. Unpublished manuscript. Århus University, Computer Science Department, Århus, Denmark.


Floyd, C. (1986). STEPS—eine Orientierung der Softwaretechnik auf sozialverträgliche Technikgestaltung [STEPS toward user-oriented system design]. In E. Riedemann, U. von Hagen, K.-D. Heß, & W. Wicke (Eds.), 10 Jahre Informatik und Gesellschaft—eine Herausforderung bleibt bestehen [10 years of informatics and society—A challenge remains] (Research Report No. 227, pp. 106-131). Dortmund: University of Dortmund, Department of Informatics. (An English version of this paper is available in the form of an internal report, Technical University of Berlin, Computer Science Department.)


---

**HCI Editorial Record.** The 1987 German report was brought to the editor's attention by Brigitte Jordan, who provided the editor with a brief translation in May 1988. Based on this, the editor encouraged the report to be translated into English and invited it to be submitted to *HCI*. First English manuscript received December 19, 1988. Reviewed by Morten Kyng and Lucy Suchman. Revision received August 17, 1989. Accepted by Thomas Moran. Final manuscript received October 2, 1989.  
— Editor
APPENDIX: POTENTIAL CONTACTS IN SCANDINAVIA

**Sweden**

Per Eric Boivie  
TCO  
Central Organization of  
Salaried Employees in Sweden  
Linnégatan 14  
Box 5252  
S-102 45 Stockholm

Hans-Eric Nissen  
Department of Information and Computer Science  
Lund University  
Sölvegatan 14 A  
S-223 62 Lund

Göran Goldkuhl  
Department of Computer Science  
Linköping University  
S-581 83 Linköping

Jerker Lundequist  
Department of Design/Architecture  
The Royal Institute of Technology  
S-100 44 Stockholm

Mats Lundeberg  
The Stockholm Business School  
P.O. Box 6501  
S-113 83 Stockholm

Peter Inström  
Bernd Hofmayer  
Swedish Center for Working Life  
P.O. Box 5606  
S-114 86 Stockholm

Peter Docherty  
Swedish Work Environment Fund  
P.O. Box 1122  
S-111 81 Stockholm

**Denmark**

Niels Bjørn-Andersen  
Information Systems Research Group  
The Copenhagen School of Economics and Business Administration  
Moritzvej 60  
DK-2000 Frederiksberg

Lars Mathiassen  
Institute of Electronic Systems  
Ålborg University Center  
Strandvejen 19  
DK-9000 Ålborg

Andreas Munk-Madsen  
Morten Kyng  
John Kammersgaard  
Computer Science Department  
Arhus University  
DK-8000 Århus

**Norway**

TomChr. Pape  
Kari Thoresen  
Norwegian Computing Center  
Forskningsveien 1 B  
Blindern  
N-Oslo 3

Kristen Nygaard  
Gro Bjerknes  
Tone Bratteteig  
Oslo University  
Blindern  
Box 1080  
N-Oslo 3