ISSN 0105-8517

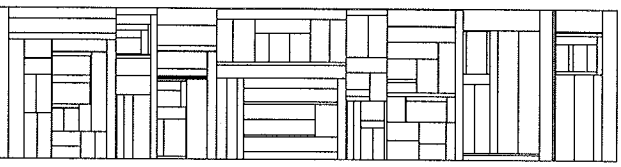
Improving Conditions for Cooperative System Design

– shifting from a product to a process focus

Kaj Grønbæk
Jonathan Grudin
Susanne Bødker
Liam Bannon

DAIMI PB – 331
September 1990

COMPUTER SCIENCE DEPARTMENT
AARHUS UNIVERSITY
Ny Munkegade, Building 540
DK-8000 Aarhus C, Denmark



Improving Conditions for Cooperative System Design - shifting from a product to a process focus†

by

Kaj Grønbæk, Jonathan Grudin,
Susanne Bødker, and Liam Bannon

Abstract

This paper deals with the conditions for cooperation between users and developers in systems development projects. At first glance, many projects seem to present immense obstacles to user involvement. At the same time, there is a growing recognition of the need for user-developer cooperation, and research projects are providing new tools and techniques that engage users as full participants in system development. Two disparate projects serve as examples to frame a discussion of the realities of user involvement (or lack thereof) in development projects. This allows us to note both the possibilities for, and the obstacles to, user participation. We believe that cooperative systems design is needed to improve the quality of interactive computer applications, and that often it can be brought about even in the face of admitted obstacles. To achieve this, users need to be involved early in the whole process, and contracts governing development may need to be re-thought: inflexibility hinders iterative design, independent of the type of project under consideration. Development contracts should be shaped as process contracts between user and development organizations with scheduled renegotiation points. In general, we believe that the concern for quality products and processes requires that systems development assume more of a process focus than is currently evident.

Table of Contents

1. Introduction 2
2. Case 1: The Advanced Workstation Project 4
3. Case 2: The School Administration project 7
4. Learning from our Cases: Creating better conditions for cooperation...11
    4.1 Late identification of partners is an obstacle to cooperation.....11
    4.2 Fixed contracts are obstacles to iterative design 13
    4.3 A product focus is too narrow 14
5. Concluding remarks.....15
References 15

† Forthcoming in Aki Namioka & Douglas Schuler (eds.) *Participatory Design*, Lawrence Earlbaum Associates.

1. Introduction

In the first decades of computer system development, most users of computer systems were engineers and programmers, so "user participation" in design was not actively sought -- the developers themselves were good user representatives. In the past 15 years this has changed substantially, as computer use spread to work environments very unlike the engineering environment. The new divisions of responsibility and the divergence of qualifications has widened the gulf between the developer and user environments, creating the need for user involvement in design.

User involvement, or cooperation between designers and users, requires tools and techniques that engage users as full participants in system development (Greenbaum and Kyng, in press). Motives range from simple cost-benefit arguments -- early involvement of users with the future system may lead to adjustment of their expectations, making eventual acceptance more likely -- to concern for working life democracy. In our view, user involvement is needed for quality reasons: To make better design *products*, the designers need knowledge about the use situation that can only be obtained through cooperation with the users (Ehn and Kyng, 1984; Ehn, 1988; Bødker, in press). This, in turn, entails focus on the quality of the design *process*. Although user involvement may take various forms, we believe that in all such situations the users must be taken seriously -- involvement cannot just be a matter of "extracting knowledge from their heads" or "adjusting their expectations." It is essential to commit to ensuring benefit for both users and designers.

Good intentions on the part of the developers have not proven to be enough to ensure successful cooperation. Structures and processes in a development organization have a large impact on the conditions for user participation. Computer systems development takes place in a wide range of contexts, each with specific advantages and disadvantages for successfully engaging current or potential system users. One commonality is that most development projects are organized around a product to be delivered -- they share a "product focus." In this paper, we argue for shifting to a "process focus," wherein the process is essential in enhancing the conditions for user involvement. In the absence of such a shift, individual developers may find themselves engaged in an uphill struggle within their organizations to obtain useful user participation and thus project success.

Systems development does not occur in a vacuum. It is a process that occurs within a particular socio-economic system. The rules and practices existing within this society play a role in defining the very framework for systems development projects. Many countries have enacted legislation requiring representatives of different interests to be involved in these projects, or management and labor have themselves made agreements on consultative processes required when changes in working conditions are proposed. In most European countries there exist formal technology agreements between trade unions and employer organizations, which establish requirements for development processes. For example, Danish employers initiating a large system development project must first obtain acceptance from the employees on the overall project goals and the expected consequences for work organization. As another example, Danish legislation places restrictions on software products that use databases containing personal information. Other societal conditions and processes include the availability and qualifications of workers -

- both current and prospective employees -- and the technology that exists in, or could be acquired by, user and development organizations. General economic conditions, including customer, supplier, and competitor relationships, are also important. Closer to the development project, the structure and processes of development and user organizations create limits and possibilities for distributing power and responsibility among different parties (e.g., managers of different departments, individual workers, and unions). As noted by Mathiassen (1981), these conditions are not static. They are changed by organizational and societal processes, possibly including the very development project itself, for example when the development organization hires people with different qualifications, or when an application that enables information collection from sensitive databases leads to political legislation governing its use.

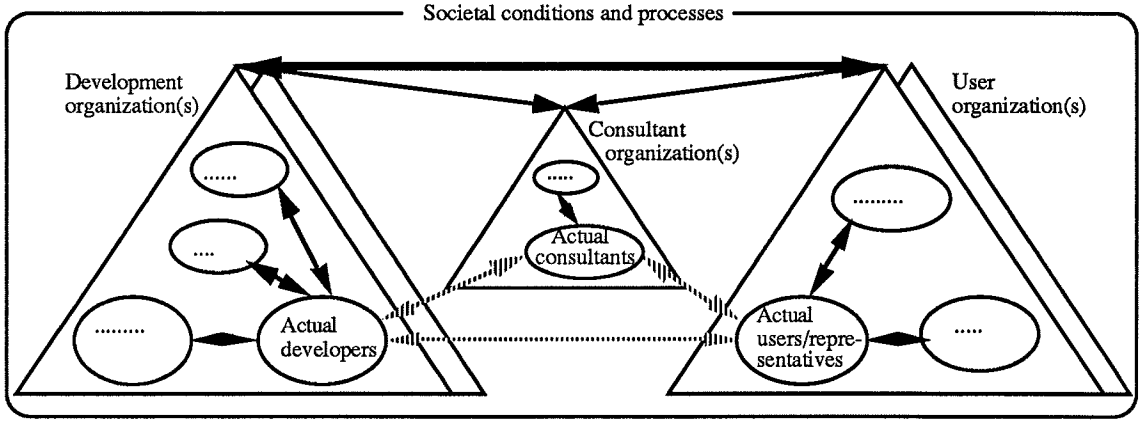


Figure 1: Prototypical partners and relationships in a system development project

These and other conditions create a diversity of situations that can help or hinder successful cooperation of users and developers. Our discussion focuses on systems development taking place in *development* projects that are necessarily limited in time and resources. (Exceptions to this kind of organization include ongoing maintenance and support of existing systems, end-user programming, etc.). In this paper, two case studies of development projects with very different contexts and outcomes are used to illuminate this diversity. These development projects are discussed in terms of the partners involved and accompanying agreements or commitments. We have diagrammed some of the possible partners and their prototypical relationships in Figure 1. We will utilize this scheme when discussing our two project cases further. The rounded rectangle represents the space of social conditions and processes in which the organizations and the project live. The triangles represent organizations, with overlapping triangles denoting multiple organizations of the same type. The primary relationships between actors are denoted by heavy lines, with bi-directionality implying mutual or reciprocal influence. Dotted lines indicate linkages that are possible but not always present. In fact, the dotted lines represent linkages that we, in this paper, strongly advocate that development projects establish. The actual groups involved in negotiations between and within each organization are shown in the ovals.

In the balance of the paper, we discuss several issues that affect successful cooperation:

- the identification and timing of involvement of the parties to a development project
- the nature of the contracts or agreements governing development
- how a process focus might be more insightful than the more typical product focus..

2. Case 1: The Advanced Workstation Project

Note: In this first case there was *virtually no actual user involvement in the design process*. In our analysis of the project we shall discuss the obstacles to and reasons for this lack of involvement. Many of the apparent obstacles point to a lack of understanding of the *rationale behind* user participation, as well as to organizational problems in achieving it.

In 1985, a large product development company assembled an elite team of software engineers to develop the company's first powerful, bit-mapped workstation to support "office automation." This project was unusual in having a relatively free design charter and schedule, whereas more typical projects were revisions of existing products, implementing under a tight schedule design changes specified well in advance by Marketing representatives. The system was to support word processing, a spreadsheet, business graphics, time management, electronic mail, and other applications to be specified over time. The application mix was part of a marketing strategy decided upon in the Marketing Division and at high levels in the company. An important aspect of the project was the human-computer interface, which the development team was to establish for both the system-level functions and the applications, which would be written by other groups.

The development team had no prior experience in involving users in the design process, apart from informal discussions with developers and secretaries who used the company's products. The typical project provides little time for such activities, few means to accomplish them (the nearest large user organizations were many miles away), and little incentive, since the changes to be implemented were specified in advance. This project, however, had both a need and an opportunity to break with that practice. Deciding between proprietary and commercially available operating systems and designing a graphical interface (new and not well understood at the time) are examples of issues that could have benefited from collaboration with prospective users. This project had substantial time and resources. But the concept of such collaboration in design was unfamiliar and was simply never considered. Let us analyze the structures and processes found in this case more closely.

Structure

An outline of some of the groups involved in the project and their interrelations are shown in Figure 2. On the left are represented relationships among groups within the product development organization. The arrows from groups in the development organization to groups in user organizations represent ties to users of existing products. Similar ties would be expected to form for the Advanced Workstation *following its release*. The software engineering group was the hub of activities on this project, not surprisingly. A User Group consisting of representatives of major purchasers of the company's products is represented by the dotted oval on the right.

In recognition of the importance of usability, a "User Interface Group" of software engineers was established. Writing a "Users' Manual" as a

preliminary design specification was an early priority. They equipped a room to support prototyping experiments. Recognizing the interlocking aspects of the software, documentation, and training, they obtained two Technical Writers from the Publications Department, provided space for two Performance Analysts, and hired someone to design the training and form a bridge to the Training Department. The Marketing representative overseeing the project attended meetings from the start. One member of the User Interface Group was assigned to be liaison to the Human Factors group; he also began working to bring in graphic design expertise from an Industrial Design group.

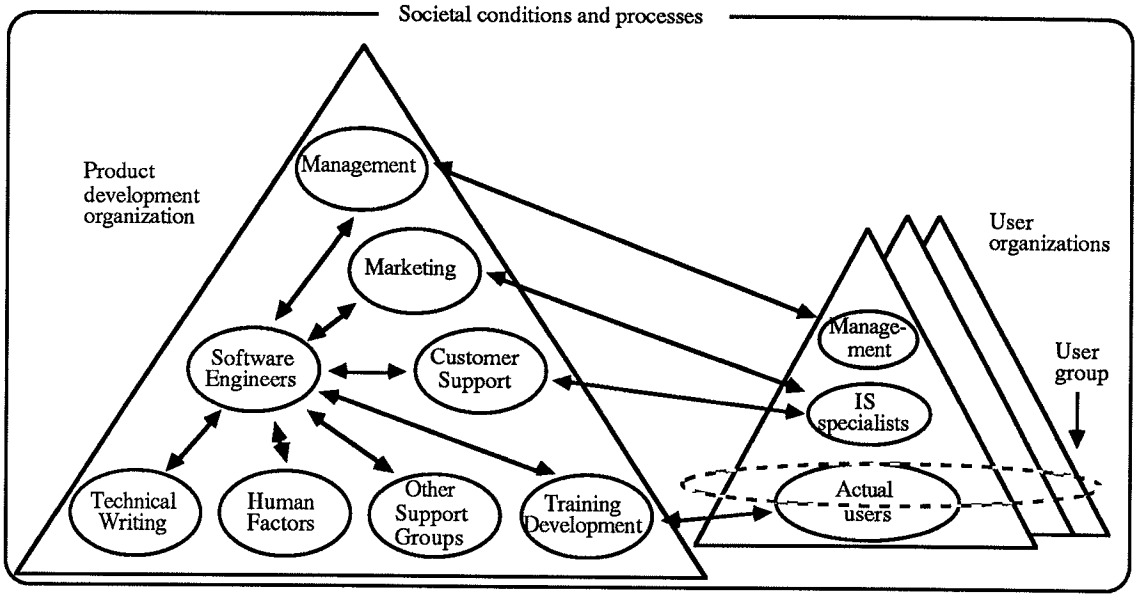


Figure 2: Project partners and their relationships in the Advanced Workstation project (Case 1).

The core project development team had only a very general sense of the future users -- office workers. Responsibility for thinking about users was sharply divided: management considered the strategic market segment to pursue; Marketing focused on application-level functionality; the development team was responsible for the system's "look and feel"; and support groups were responsible for documentation, training, and other aspects of the system. In fact, the preliminary "Users' Manual" was actually used to communicate the interface design among these groups and was never reviewed by actual office workers.

Process

This project organization did not survive organizational pressure. The Technical Writers felt their career paths jeopardized by their separation from the Publications Department and returned to it, assuming a more distant involvement in the project. The Training Developer found that senior people in the geographically distant Training Department were unwilling to yield the overall design to her, and eventually she also left. The Performance Analysts produced a list of operations and minimum response time requirements for the product that the developers found unmotivated and difficult to translate into meaningful design constraints; the Performance Analysts soon stopped attending meetings. The effort to involve graphic design specialists in designing displays and icons took a full year, due to the need to educate them about the software and to overcome the resistance of some software engineers,

who while inexperienced had grown fond of their own designs. The principal Marketing representative focused on the application set, so marketing input to the user interface came sporadically from encounters with design specifications. For example, someone from International Marketing would appear and demand that features be changed. The Human Factors group carried out experiments on input device design, windowing techniques, cursor control keys, and other aspects of the system, using internal employees (secretaries or developers). Their data sometimes helped resolve ongoing arguments, but typically were seen as covering too little and arriving too late. As developers grew increasingly loathe to "throw away code," a separate prototyping group was established. This worked well until a crunch came, at which point they were drafted into development and the prototyping effort stalled.

Contacts and agreements with external groups were minimal. At various times the project received high-level advice from industry consultants and IS specialists in major corporations by showing them the specifications and by bringing them in for round-table discussions ("focus groups"). At one point, Marketing arranged with management at a few major customer sites to allow several developers to visit clients. The developers were astonished at how little the company's products in use resembled those leaving the development company. Customers did considerable modification, adding software or altering the system software so that most users would see only a fraction of the functions, for example. After over a year of development, the company provided a major client with several "prototype systems" consisting of hardware and the most basic system software, entirely for use by the client's in-house development groups. During the course of the project, a major Users' Group meeting was held nearby, but developers were denied permission to attend. Marketing considered it to be a show for customers.

The project continued for three years and was ultimately terminated. Development had produced working prototypes of the system (though not of all applications); the concern seemed to be the lack of a market for the product.

Summary and discussion

As we mentioned at the outset, this project did not directly involve prospective users in the design process at all. To some extent, developers regarded themselves as prospective users. This was true in that the company had a policy of using its own systems in the development lab, but the developers were not representative users in many respects -- it was recognizing this fact that shocked the developers who saw the modified systems at customers sites. In this environment, Marketing representatives, Human Factors Engineers, Industrial Design Engineers, Technical Writers, Training Developers, and Performance Analysts often describe themselves as *spokespeople for the users*! In that sense, this project was unusually good at recognizing the need to include these perspectives from the start of the project. Members of these support groups often feel that they become involved on projects too late (Grudin and Poltrock, 1989). In an environment where such indirect "users' representatives" often find it difficult to be involved, it is small wonder that no one even contemplates involving actual users!

Would cooperation with potential users have enabled the project to avoid the problems that overtook it? Identifying prospective users is difficult in such a project, something which is often given as a reason for not doing so. However, finding some plausible set of users might have provided a useful, common yardstick for the entire project, and indeed we think that finding such users is

possible. A group of users representing some aspects of the prospective use practice seems to be a better choice than none (it is better to have some idea about how the application can be used than none). Resources for this project were quite substantial, and the available resources could have supported partial salaries for such users for the time they would need to devote to the project. Furthermore, many user organizations, interested in gaining state-of-the-art experience with technology, are open to cooperation. It is quite conceivable that such projects could enlist active user involvement in cooperative design. For an example of a development project that involved users in a complex and interesting way, let us now look at another case, and then make some comments on similarities and differences re. possibilities for user involvement.

3. Case 2: The School Administration project

In our second case, both users and designers accepted the need for good relations and worked for the establishment of good conditions for cooperative design. The project concerns a long-term (4 year) effort to develop an administrative system for approximately 100 schools (User organizations). The description is based on direct contacts with developers in the Software Company and also on the results of 2 empirical projects undertaken at Aarhus University - Klujeff, Møller, and Petersen (1989) and Holk, Lauridsen, and Nielsen (1989). The estimated number of end-users was 3000. The project started with competitive bidding on a fixed-price product contract. But circumstances led to renegotiation of both requirements and price in a way that resembles an in-house development project. Let us again discuss the project using the framework of structure and processes found on the project.

Structure

The system consisted of ten sub-systems to be developed partially in parallel. The system included student administration, building administration, etc. The contract was awarded by the Department of Schools (the customer) to a hardware vendor, who subcontracted the software development. The Department hired a consulting company to assist in the project (e.g., by managing the tests). Figure 3 is an overview of the partners and relationships according to our scheme. For instance, the heavy horizontal arrow represent the original fixed-price contract between the Customer and the Hardware Company. The arrow linking the management of the Hardware and Software Companies represents a fixed-price subcontract between them. The arrow linking the software engineers in the Hardware and Software Companies represents a more informal communication structure to provide mutual support.

The hardware company (together with the software company as subcontractor) had won the competitive bidding with a system sketch defining the sub-system division. The contract provided for a fixed amount to be paid to the hardware company, which was responsible for selecting the hardware and specifying the user interface for the software company. A fixed price ($1.24 million) contract for software was also established, some of it directly negotiated between the Department of Schools and the Software Company.

The experience of individual systems developers with both the technology and the application domain varied. Some had experience in trade school settings, others did not. In fact, one developer with experience in the application domain was hired for this project

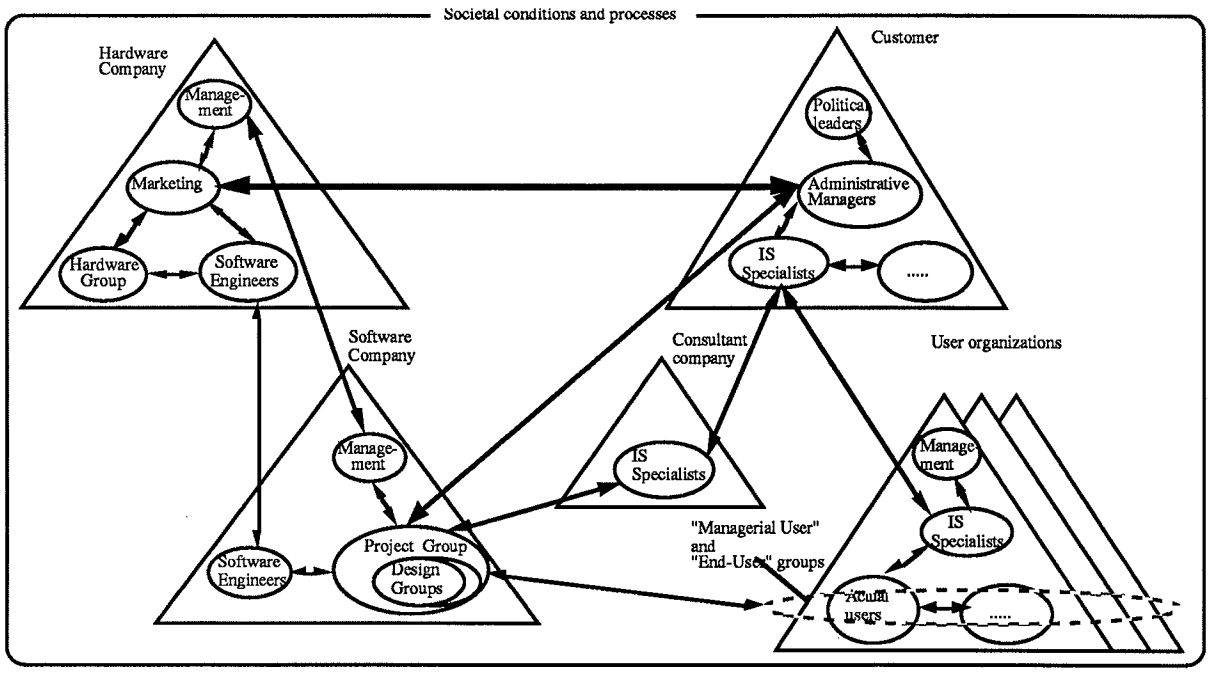


Figure 3: The School Administration Project : Partners and their relationships in (Case 2).

The software developers incorporated their own ideas about the development process into the plans. For example, they insisted that the contract contain a provision specifying the development of prototypes.

The initial conditions of the project contained an agreement specifying the involvement of users and their organizations in the project. In addition to their role in the actual computer system design, the agreement contained provisions for user training. A "Managerial-User" group with management level users from 6 schools was formed and contributed to the overall requirements specification and prototype testing. Some of the Managerial Users had previous experience with a system development project. An "End-User" group consisting of representative actual users from the schools was established: They were to test prototypes and be educated to teach the use of the sub-systems to the large group of actual users at each site. The "End-User" and "Managerial User" groups got economic compensation for their project work: 400 hours of project work annually per person, which roughly corresponded to the number of hours actually spent.

Process

In the beginning the system developers and the Managerial Users worked on refining the requirements specifications for all the sub-systems. This specification process incorporated the use of paper mock-ups of screen images and the development of an initial prototype. However, when the End-Users and the Consultants were involved in testing the prototype, it became clear that these were incomplete and out of touch with the demands of the actual users. This emerged during the development of the first sub-system. Here, a prototype was developed exclusively based on the initial requirements specifications. During prototype test, the Consultants and the End-Users faced so severe problems with the design that they demanded more influence on the design as a condition for continuing to participate in the project. At this point the project was in serious trouble: The user representatives would refuse to use

a system developed according to the specification and initial prototype. At the same time the Software Company claimed that major changes of the specification would require renegotiation of the contract. The customer realized that it was partly their fault that the initial specification was out of touch with the actual work carried out in the user organizations. In order not to lose the initial investment, the Customer renegotiated the original product contract with the Project Group (the negotiation is represented as the diagonal arrow between the Project Group and the Administrative Managers in Fig. 3). In these negotiations the Software Company proposed to establish a *process contract*, explained below.

Following these negotiations the project was able to move forward under a new project model, based on a process contract: For each sub-system, a design group was created to produce a new requirements specification. A design group typically consisted of two systems developers, one representative from the consulting company, one Managerial user and 2-3 actual users. The sub-systems were developed according to the plan of Figure 4.

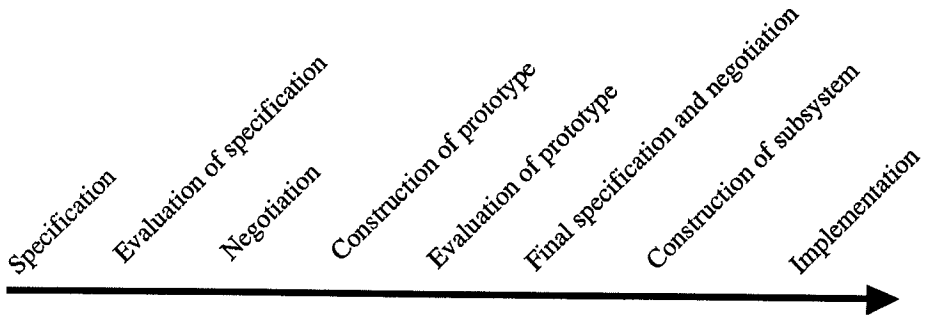


Figure 4: Plan for each sub-system

The design group worked from the initial requirements specification to produce a new requirements specification for the specific sub-system. The Managerial Users evaluated the requirements specification and assessed the changes to the initial requirements specification -- those requiring modification of the contract. New requirements considered to be vital were included in the sub-system requirements specification; less important requirements were set aside for possible future development. The cost of these changes was considered in reassessing the price. The software company constructed a prototype, with a users' guide, covering 80% of the functionality. The prototype was tested by the involved users, some of the Managerial Users, and the consultants in a two-day test based on the requirements specification and the users' guide. On the basis of this test, the requirements were again modified and prioritized and the price renegotiated. The software company then constructed the sub-system, which was tested, corrected and given a final implementation. Thus, the plan included two renegotiations of requirements and price for each sub-system. Experiences from prototype evaluation contributed to an incremental development process, as the implementation of sub-systems proceeded. End-Users participated in the design groups by producing requirements specifications, evaluating changes to them, and by testing prototypes.

All of the sub-systems were developed and the four-year project came to a close only $2^{1/2}$ months later than planned. Starting from the original specification of

the system functionality, 1800 changes were proposed by the users, extending the functionality of the system approximately 50%. 350 of the proposals were given low priority and postponed to a follow up project. The total price of the system was increased by $0.6 million, but the users got a usable system to support their work.

Summary and discussion

The initial requirements specification was provided by the customer and refined by systems developers and management-level Managerial Users. This procedure was giving the Managerial Users considerable opportunity to influence the requirements. However, they did not have a sufficiently detailed knowledge of the actual work carried out at the Schools to be able to specify a usable system. Thus, the prototype of the first sub-system was rejected by the end-users. The rejection led to a renegotiation of the contract changing it into a contract focusing more on the development process. This led to the creation of design groups with end-user participants and an iterative prototyping approach. In the design groups actual End-Users assisted the Managerial Users in revising requirements specifications in an environment with few constraints imposed by contracts or resources. This way, actual users influenced the system design. The Managerial Users, though, could later remove or postpone certain features for economic reasons. The consultants initially were to participate only by testing the prototypes and sub-systems. As a consequence they were to influence the design of the system through reports of errors and missing features. However, when the first sub-system turned out to be a failure, they objected to this process. Together with the End-Users, they succeeded in influencing the structure of the project organization, because the Customer recognized that the problems were due to their initial faulty and incomplete specification.

The development of the later sub-systems seemed to run more smoothly. The involvement of users in early specification activities worked well in one respect: The Managerial Users had a good feeling for the overall sub-system requirements, but lacked knowledge of the daily work situation. The heavy reliance on Managerial Users in the first requirements specification phase may explain why testing of the initial prototype resulted in so many suggestions for change. For example, the paper mock-ups of screen images were not provided to end-users to relate them to the future use situation. Through subsequent cooperation with actual users, the systems developers overcame this problem and obtained a setting that provided the feedback they needed to develop a good system. In all, 64 (out of 3000) End-Users from 32 of the Schools participated in the design activities and thus contributed actively to the design of the system.

4. Learning from our Cases: Creating better conditions for cooperation

Our two cases demonstrate various problems in establishing proper conditions for involving users in system development. In the first case, we note the following points: the future users were poorly defined, there was no tradition of direct user involvement in design, and responsibility for thinking about the users was divided. In the second case, the initial emphasis on management-level users appeared to be a mistake due to their lack of understanding of the actual work processes. The original product contract constituted a barrier to having direct end-user influence on the design. In this section we discuss

development conditions and possible ways to overcome the barriers to cooperating with users in these kinds of projects. Some development contexts may profit by borrowing techniques that have evolved more naturally in another context. We will focus the discussion on several issues that we found of particular importance:

1) the identification of users to be involved,

2) the timing of their involvement,

3) the contracts and agreements governing a project, and

4) the product focus of most projects.

4.1 Late identification of partners is an obstacle to cooperation

The times at which different potential partners in development are identified -- the developers, consultants, and users -- are factors requiring careful consideration in planning for greater user participation in design. The best prospects for a balanced cooperation would logically occur when both developers and users are identified at the outset, as is often true for in-house development projects. For example, a large bank decides that its internal software development group will develop a system connecting tellers and platform managers. Such projects are, however, not without challenges, and neither are other projects where both developers and users have been identified, as in Case 2. Choosing representative users is not easy -- within a single organization one finds groups with conflicting interests, conflicting perspectives on the organization, and with very different possibilities of exercising power over one another (Sandberg, 1975, Ehn & Sandberg, 1978, Ehn, 1988). These projects may have few resources to spare. Cooperation may inadvertently antagonize users, not least if they see that their own "participation" is mainly a way in which management and designers "extract the knowledge from their heads" without them gaining any real influence on their own working conditions in return. A similar danger is of raising their expectations unduly, diminishing the likelihood of system acceptance. Nevertheless, the early identification of all parties is a favorable condition for cooperation.

The product development environment discussed in Case 1 is quite different. In a "pure" product development situation, the eventual users are not discovered until development is complete. The initial product idea does include *some* conception of a user population, but there is little pressure to produce a detailed description. Specific thoughts may even be closely guarded by management to prevent it from reaching competitors. In fact, a vague description may seem to support the desire to appeal to as large a market as possible. This mind-set of inclusiveness runs counter to the idea of relying heavily on a handful of users.

For these reasons, in environments in which the development team is clearly identified but user organizations are not, a product focus creates substantial difficulties for full user participation. Even the support groups that see themselves as user advocates, such as technical writing and human factors, are often denied early involvement, due to the perception that development efforts nearer to completion have more urgent need for their participation. Overcoming these obstacles will require changing the development approach to focus on the process, in order to create conditions for user involvement that do not emerge naturally or easily.

Even with a commitment to user participation, identifying the prospective users and the "team" with which they are to collaborate is not easy in a product

development environment. Information about existing and targeted user populations is distributed among people in management, marketing, sales, and customer support organizations. The development team is often a distributed group, with members from software, human factors, technical writing, training development, and other groups involved at various times and with varying degrees of communication. On the positive side, these organizations often have considerable motivation to improve the usefulness and usability of their products, the resources to commit to it, and large numbers of prospective users. (These favorable and unfavorable conditions are covered in more detail in the chapter by Grudin.)

The Scandinavian UTOPIA project (Ehn and Kyng 1984; Bødker et al.,1985; Bødker et al., 1987) was a research project that applied techniques more typically used in in-house development projects to the design of a system intended for broader distribution. The typographers selected by the graphical unions to cooperate with the researchers were chosen not because they were likely to become actual users of the system, but due to their professional skills and political interest. The typographers continued working part-time at their newspaper to avoid cooption into the technical team and to obtain feedback from a wider group of typographers. Furthermore, the project used newsletters and seminars to reach additional typographers, journalists, newspaper managers, and researchers. Such approaches might be used in commercial settings to explore the possibilities of specific application areas, narrow or wide, and to provide feedback much earlier than is available through alpha or beta site testing, although sensitivity about disclosing plans may be an inhibiting factor.

A different problem with the timing of involvement arises when development follows competitive bidding. Here, too, we find a very natural product focus: the requirement to specify the system to be built. In this case, the customer/user organization is first involved. It may define the system without input from the eventual developers, although perhaps with the help of consultants, as in our second case study. The developers may not be able to modify or "break" the contract as part of an interactive design process, and may even be discouraged or prohibited from establishing ties to the user organization for reasons of security or for fear of creating ties that would represent favoritism on subsequent contract bidding. In our second case the initial contract terms were modified substantially due to problems that occurred with the initial product specification, but contractual arrangements are in general more inflexible.

For large competitively bid contracts, the motivation and resources to overcome these obstacles may exist. The next section discusses a key element in overcoming the obstacles -- the development of more flexible, process-oriented contracts.

4.2 Fixed contracts are obstacles to iterative design

In Case 2, competitive bidding led to a fixed product contract between the development organizations and the customer/user organizations. The contract assumed that the development of the system could be based on an initial specification made by a group of management level users together with the developers, but, as described in section 3, the customer realized the need for renegotiating this in order not to lose the initial investment. Thus, it was possible to set up a new project model that included price renegotiation, iterative design, and incremental delivery of sub-systems. Some of the new features that emerged as a result of direct user involvement would be expensive to implement, so it was agreed to renegotiate the price based on assessment of

specific features as they appeared in prototypes developed by the design groups. This case suggests that an initially disastrous design proposal can evolve into a successful development project when a fixed contract is replaced with a strategy providing for flexible renegotiation.

The issues raised by fixed contract proposals are too complex to take up fully in this paper. In some sectors, changing socio-economic conditions may well contribute to an increase in the number of such competitive bidding proposals. In discussing problems for cooperative system development in contract situations, Gundry (1988) claims that "co-operative system acquisition" is losing ground to competitive purchasing due to the difficulties of controlling money and time with the former policy. He is pessimistic about the possibility for extending the flexibility of contracts and argues that we must try to find a way to fit user concerns into the more restrictive competitive type contracts, although he recognizes the difficulties and limitations in doing so. We do not share his pessimism about more cooperative strategies between users and developers and more flexible process type contracts. In our own experience, we have seen encouraging signs within the commercial sector that more flexible process type contracts can be negotiated successfully.

In general, fixed contracts may hinder iterative design, independent of the type of project considered. Development contracts should be shaped as process contracts between costumer and development organizations with scheduled negotiation activities. Where possible, contracts might incorporate aspects of the project model eventually used in the second case. Boehm (1988) proposes that in-house development projects incorporate iterative design through cyclic or spiral models that explicitly schedule checkpoints for considering the renegotiation of future activities. For contract development projects, he describes experimental approaches to providing more flexibility, including multi-stage contracts in which several developers produce prototypes in the initial round, level-of-effort and award-fee contracts for evolutionary development, and design-to-cost contracts. Project models such as the spiral model seldom pay explicit attention to the degree of user involvement. The opportunity for user involvement is often hidden under the label of phases such as "prototype evaluation" or "test." However, we suggest that more active user involvement, at least to the extent seen in Case 2, is more likely to produce usable interactive systems. Thus, contracts between development and user organizations could also provide explicitly for the use of certain cooperative design techniques (see e.g., Greenbaum and Kyng, in press; Bødker and Grønbæk, 1989) to facilitate active user involvement at different stages. These suggestions for contract formation generally put a greater emphasis on process issues.

4.3 A product focus is too narrow

Cases 1 and 2 deal with development tasks where the user and development organizations are separate units with no history of cooperation. The projects start out trying to specify the product to be developed. In this sense they are very similar to traditional system development projects that start by attempting a *complete* product specification, with the inherent assumption that the future users' needs can be completely analyzed and anticipated. This is a strategy, however, that leads to many products being rejected by users (Ehn, 1988; Bødker, in press). The potential magnitude of this problem is reported by a U.S. General Accounting Office study of nine software development projects that found that under 5% of the money spent resulted in software that could be used as delivered or with minor changes (Martin, 1988, p.4).

Both case 1 and 2, however, differed from the traditional approach to varying degrees, recognizing a greater need for focusing at the outset on the needs of potential or actual users. Case 1, the workstation development project, provided only a modest change of focus, primarily through greater communication among those responsible for different aspects of the user interface. In contrast a more radical shift of focus is seen in Case 2, the school administration project. The development project is changed from a fixed contract for the delivery of a piece of pre-specified software to an evolutionary design process with continual, active involvement of actual users. Time is invested in establishing coordination patterns that would not have formed without the partners explicitly attending to the process and the conditions required for successful cooperation. In Case 2, this investment in the process as such is the main reason for the successful outcome of the project.

There are encouraging signs of a shift away from a simple product focus toward an understanding of the process of system design, development, use, redesign, and reuse. Breaking down (or tightly coupling) the development/use distinction requires re-thinking the systems development process and the nature of the commitments involved. A number of voices, from differing backgrounds, can be heard in support of this shift. For example, the designer Chris Jones (1988) criticizes the tendency among software designers to take the product as central and the users only in relation to their use of the fixed product. He argues for more attention to the process as an end in itself, not simply a means. Regarding the standard procedure of building to a fixed set of specifications, he notes: "Both parties (designers and clients) have to give up the use of the requirements as a semi-legal basis for control and measurement and agree to work together in the continuous meta-process of evolving the brief and sharing in the eventual decision as to how the problem is to be seen and solved.". Continuing, he notes: "Functions, statements of requirements, are essential but temporary. Without them we cannot begin, but unless we can change them we cannot finish, cannot discover .....recognize that the "right" requirements are in principle unknowable by users, customers, or designers at the start." This position calls into question the nature of most formal contracts today. As another example, the consultant Tom Gilb (1990) stresses the need to focus on process, not method or static product. Critiquing current development "methodologies," he notes: "They are based on a static product model. They do not adequately consider our work to be a continuous process - derived from the past and being maintained into the future". From academic software engineering, yet another voice supporting this shift is that of Floyd (1987). She argues for more emphasis on the process of software development than on the efficiency of the resulting code: "The product-oriented perspective regards software as a product standing on its own, consisting of a set of programs and related defining texts... considers the usage context of the product to be fixed and well understood, thus allowing software requirements to be determined in advance," while the process-oriented perspective "views software in connection with human learning, work and communication, taking place in an evolving world with changing needs... the actual product is perceived as emerging from the totality of interleaved processes of analysis, design, implementation, evaluation and feedback, carried out by different groups of people involved in system development in various roles." This shift in perspective calls into question the separability of development and use and the concept of a "complete" requirements specification. Again, Gilb (1990) admonishes software developers to accept that requirements are always changing and to focus on handling this situation by catching problems early through user

involvement in the design and specification phase. He also stresses the need for prototyping and frequent iteration. If we accept this view, the significance of the obstacles in many existing development contexts is apparent. In contexts lacking inherent patterns of cooperation, a deliberate effort is required to change the conditions, but there is a good chance that the effort will pay off.

5. Concluding remarks

We have described obstacles to user involvement in design that arise in development contexts that are widespread today. Effort and expense will be required to find and implement techniques to overcome them -- to develop methods and tools, and to change organizational practices and perspectives. We have argued that the concern for quality products and processes is the driving force: to create better computer applications, developers need to be more concerned with the work processes in which the computer systems will eventually be used and with the outcome of these processes. Many traditional ways to obtain knowledge about work processes have failed (see Bannon & Bødker, 1989; Bødker, in press). We suggest greater emphasis on user-developer cooperation in the development process. Such cooperation requires early identification of the project partners and flexible contracts. The focus of contracts should be on creating a development process based on user-developer cooperation rather than attempting to specify the product fully in advance. In addition, many of the costs of providing a more usable system are decreasing -- computer processing time, memory, maintenance, and so forth. Thus, both market pressures and concern for the use of technology on behalf of those working with it are leading in the same direction. We have discussed apparent obstacles to these positive changes and attempts to overcome some of them. Although these attempts were not uniformly successful, we see realistic approaches that show promise for most system development projects.

References

Bannon, L., & Bødker, S. (1990). Beyond the interface: Encountering artifacts in use. In J. Carroll. (Ed.), *Designing Interaction: Psychological Theory at the Human-Computer Interface* .

Boehm, B. (1988). A spiral model of software development and enhancement. *IEEE Computer, 21,* 5, 61-72.

Bødker, S. (in press). *Through the Interface – a Human Activity Approach to User Interface Design.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Bødker, S., & Grønbæk, K. (1989). Cooperative prototyping experiments: Users and designers envision a dentist case record system, in J. Bowers & S. Benford (Eds.), *Proceedings of the First European Conference on Computer-Supported Cooperative Work, EC-CSCW'89* (pp. 343–357). Also available as DAIMI-PB 292.

Bødker, S., Ehn, P., Kammersgaard, J., Kyng, M., & Sundblad, Y. (1987). A Utopian experience. In Bjerknes, G., Ehn, P., & Kyng, M. (Eds.) (1987) *Computers and democracy: A Scandinavian challenge.* (pp. 251–278), Aldershot, UK: Avebury.

Bødker, S., Ehn, P., Romberger, S., & Sjögren, D. (Eds.) (1985). *Graffiti 7. The UTOPIA project: An alternative in text and images.* Stockholm:Arbetslivcentrum.

Ehn, P. (1988). *Work-oriented design of computer artifacts.* Falköping: Arbetslivscentrum/Almqvist & Wiksell International.

Ehn, P., & Kyng, M. (1984). A tool perspective on design of interactive computer support for skilled workers. In M. Sääksjärvi (Ed.), *Proceedings from the Seventh Scandinavian Research Seminar on Systemeering* (pp. 211–242). Helsinki: Helsinki Business School.

Floyd, C. (1987). Outline of a Paradigm Change in Software Engineering. In G. Bjerknes, P. Ehn, & M. Kyng (Eds.), *Computers and democracy – a Scandinavian challenge,* (pp. 191-212). Aldershot, UK: Avebury.

Gilb, T. (1990) Project Management for the 1990s. *American Programmer,* (pp. 16-30.)

Greenbaum, J. & Kyng, M. (Eds.) (in press). *Design at Work: Approaches to Collaborative Design.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Grudin, J. and Poltrock, S. (1989). User interface design in large corporations: Communication and coordination across disciplines. In *Proceedings CHI'89 Human Factors in Computing Systems,* (Austin, April 30-May 4).

Gundry, A.J. (1988). Humans, computers, and contracts. In D.M. Jones and R. Winder (Eds) *People and computers IV.* Cambridge, UK: Cambridge University Press.

Holk Lauridsen, M. & Nielsen, H. B. (1989). *Designprocessen som et sprogspil - belyst gennem et empirisk studie af 3 designprojekter* [The design process as a language game; seen through an empirical investigation of three design projects]. Unpublished Masters Thesis, Aarhus University.

Jones, J.C. (1988) Softecnica . In J. Thackara (Ed.) *Design After Modernism: Beyond the Object.* London: Thames & Hudson.

Klujeff, R., Møller, E.M., & Petersen, K.V. (1989). *The role of users in systems development.* Unpublished Masters Thesis, Aarhus University.

Martin, C.F. (1988). *User-centered requirements analysis.* Englewood Cliffs, NJ: Prentice Hall.

Mathiassen, L. (1981). *Systemudvikling og systemudviklingsmetode* [Systems development and systems development method] (DAIMI PB-136). Århus: University of Aarhus.