Scandinavian Journal of Information Systems

Volume 8 | Issue 1 Article 7

1996

Computers in Context — But in Which Context?

Torbjörn Näslund
Linköping University, Sweden, tor@ida.liu.se

Computers in Context — But in *Which* Context?

Torbjörn Näslund

Department of Computer and Information ScienceLinköping University
S-581 83 Linköping, Sweden, tor@ida.liu.se

Abstract

This is a case study of industrial systems development. In the project studied, an advanced computer application was developed with the intention of supporting complex, cooperative work. The project participants all viewed the emerging computer application in context. However, different contexts were used by each group of actors. Since the application was placed in contexts well known by the participants, and within which their own expertise could be applied, this phenomenon could be seen as an advantage. However, a drawback was that the participants could easily misinterpret each other. They also restricted their actions to qualities which were important for the relation between the application and the particular context visible for each actor. In conclusion, systems development must transcend these limitations, while at the same time take advantage of the focusing effect each of the contexts provide.

Keywords: Professional systems development, industrial systems development, qualitative research, case study, design communication, context, usability, quality.

This article is to appear as a chapter in the book edited by Morten Kyng and Lars Mathiassen tentatively titled *Computers and Context: Joining Forces in Design* (MIT Press, 1997). The article is here printed prior to the final copy-editing. The article is printed with the kind permission of MIT Press.

1. Introduction

It is extremely difficult to develop advanced computer applications for the support of complex human tasks. A broad range of skills and knowledge is needed for the development process. Actors with different backgrounds and experiences must cooperate. The actors must both be able to concentrate on the task in which they are specialized, and communicate and coordinate their work with others. This may create a conflict between work practices which are suited to each particular specialist in the development, and work practices in which efficient cooperation and communication can occur. We need to find work practices for systems development which take both these needs seriously.

The participatory design community has been successful in suggesting how cooperation in systems development can be achieved. These suggestions have typically been based upon projects in which the researchers themselves have had a very active role. However, most systems development is performed by professional system developers outside academic settings. If we are to improve professional practice, we must be in a position to understand it. The case study reported in this chapter adds to out knowledge of professional systems development. Earlier examples of investigations into professional systems development are the Danish MARS- (Andersen *et al.* 1990) and ROSA-projects (Bødker & Greenbaum 1988).

This chapter highlights a specific aspect of the findings from a case study in professional systems development: The different actors all held contextual views on the computer application to be developed. However, their views were different, since they all viewed the computer application in different contexts.

2. Method

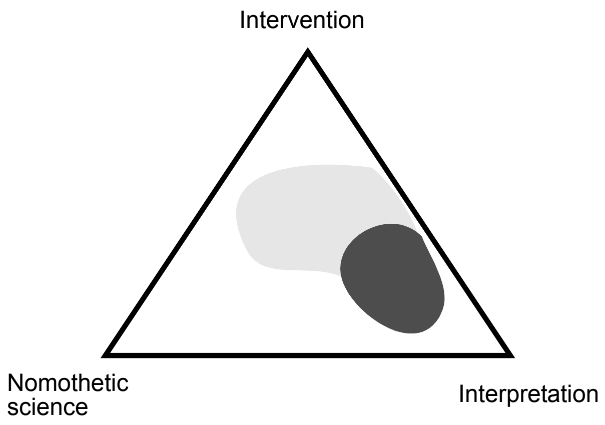
The methodological mix used for the research described in this chapter emphasizes the understanding of professional systems development. An industrial systems development project was studied in depth. One of the aims of the research project was to actively explore the effects of formative usability evaluation in professional systems development. Another, broader, aim of the project was to better understand how usability issues are handled in professional systems development. This chapter reports on the latter of these two aims.

Figure 1 illustrates the research approach of the study, using the framework introduced by Braa & Vidgen (1997). The light grey area denote the research approaches employed in the whole study. The dark grey area denote the research approach underlying the findings reported in this chapter. As illustrated, this chapter is based on an interpretative part of a case study which also included some intervention-based approaches.

I was the main investigator in the project. During the interventions, I acted as a voluntary usability evaluator.

For the work reported here, the more active role as a usability evaluator is important to know for two reasons. First, the reader should be aware that when "the usability evaluator" is described, I am actually objectifying myself. In my role as qualitative researcher, I am describing myself in my more active role. Secondly, the reader should be aware

FIGURE 1. The research approach of the study



that the active role is important for achieving the needed level of preunderstanding and access needed for collecting data and making interpretations (Gummesson 1988). The role as a usability evaluator provided me with very close access to the project and most of the project participants. Despite the fact that it was common knowledge that I, as a researcher, collected qualitative data, my presence in the project was perceived as natural due to the active role I played as usability evaluator. The need to collect information for my research and for performing the formative evaluations coincided to a large extent. I provided feedback of design suggestions and recommendations regarding usability, but did not participate in the project as a decision maker in regards to design.

In the terminology used by Blomberg *et al.* (1993), my role can be characterized as a *participant observer* regarding issues closely related to usability, and an *observer participant* regarding other issues in the project. The work reported here is written from a observer participant point of view.

I took field notes during meetings and discussions, and expanded these into a diary during breaks and evenings. Furthermore, I made formal and informal interviews with other participants. In particular, I want to highlight the value of informal interviews during lunch breaks. During lunch breaks, it was easy to make the participants explain for me—the researcher from 'outside'—how they viewed meetings in which we had participated earlier during the day. Together, the observations made during meetings and the information given during lunch and coffee breaks were invaluable for gaining insight into the views and values of the participants.

I participated several full days each week in the development phase called "design and prototyping", and for some odd days a week during the subsequent five months.

In addition to the studies in the development project, I made thorough observations and interviews in the work setting where the computer application would be used.

I have education in and experience from both professional systems development and from the application domain. This dual background made it easy for me to understand the representatives for both the development and the use settings.

3. The Use Setting

The computer application developed by the studied systems development process is a support system for coordinating role play during crisis management training of decision-makers.

Role play is performed in order to provide a realistic environment, in which crisis management training can take place. The particular use setting in question is the training of command units for the Swedish defence. Similar types of training can also be found for command units for other types of crisis management, e.g. command of fire brigades work in defeating wood fires, command of rescue operations at sea, or redirection of traffic in case of road blocks and traffic jams.

Role play simulates the environment for the command unit which should be trained, so that the unit officers can acquire experience from handling difficult situations. The role play requires a considerable amount of coordination and collaboration between the different role players, in order for the role play to become realistic and consistent.

In the particular use setting which was of interest in this project, the role players are skilled in military command and control, but participate as role players for only a few days. They do not work in permanent teams.

A few years ago, one military school acquired a computer-based experimental application to support the role players. Experience from the use of this computer application showed that the basic idea underlying the application was very promising. However, the problems related to handling the application had been considerably greater than expected. Demands for revising the user interface occurred early. It was identified that the computer application offered good services, but that the users were not able to handle the computer application effectively. As an intermediate solution, special operators were trained. These operators now serve as mediators between the application and the role players, so that the role players do not need to handle the user interface directly.

4. The Development Setting

The experiences from the first application resulted in the demand from two military schools to procure computer applications based on this same basic idea, but with important improvements made. Early on, it was decided that ease of handling the application should be given attention. An important objective was to avoid the involvement of specially trained operators mediating the use of the new application. The new application should be possible to use directly by the role players themselves.

The demand from the two schools led to the initiation of a systems development project. The development process was based on a contract between the Materials Administration of the Swedish

Defence (the customer) and a large systems development company (the contractor). The customer selected the contractor after a competitive tender process, based on a rather vague requirements specification. The contract contained a slightly more detailed version of the requirements specification.

The contract stated a fixed price for a total system, including hardware, software and installation. The contractor used subcontractors for parts of the project. The contract stated several different deliverables during the course of the project. Through these intermediate deliverables, the customer could get an update on the progress of the project, while the contractor received payment for completed work.

Grudin (1991a) distinguishes between three ways of organising systems development: Contract development, product development, and in-house development. In this taxonomy, the studied development project was an example of contract development. The project was far too large and technically complex to make in-house development a feasible alternative. Since the application to be developed needed to be custom-made, also product development was infeasible.

The combination of high demands for usability and contract development is particularly interesting from a research point of view. We know that cooperation between different parties in design are important for achieving usability. While we have begun to learn how to create cooperation in in-house development (Greenbaum & Kyng 1991, Schuler & Namioka 1993) and product development (Grudin 1991b, Wiklund 1994), we lack knowledge regarding cooperation in contract development. This is unfortunate, since this is a setting in which a great deal of complex cooperation occurs (Grudin 1991a, Grønbæk *et al.* 1993, Thomsen 1993).

5. Groups of Actors in the Project

The studied systems development project involved a large number of actors. Many of them were only peripherally involved. The number of main actors, who spent a substantial part of their time on the project, was approximately fifteen. These actors can be classified into five groups, with rather clear boundaries. The groups are described below.

5.1. The Officers

Two military officers took part in the entire project. They were commonly referred to as 'the users' by other actors in the project, although this reference is misleading. The primary users of the new application will be the role players. In this chapter, these two actors are instead referred to as 'the officers.'

The two officers were representatives of the two schools which would use the new application when the development process was completed. One of the officers had considerable experience in managing training sessions using the existing application, while the other had minor experience in this. Both of them had good knowledge of the application domain.

Thus, the officers had a role similar to what is sometimes called "managerial users" (e.g., Grønbæk *et al.* 1993). No "end user" (i.e., potential role player) participated in the project.

5.2. The System Developers

Most of the system developers were employed by the systems development company acting as the contractor. Additional system developers were hired by the contractor from several consultancy firms.

The typical background of the system developers was in computer science and software engineering. Their education and experience varied. System developers holding university degrees had slightly less practical experience of systems development, while those without university degrees compensated this with more practical experience. My impression is that the level of skills and experience in the team was what one can expect to find among software development professionals.

The team of system developers was temporarily composed for this project. Since many of them worked for the same company, many of them knew each other, but they did not constitute a permanent team. System developers were added and removed from the team during the course of the project. This was the common way of establishing project groups within the company.

One of the developers was the head of the group. Her role was called 'the program project leader.'

5.3. The Two Project Management Groups

There were two groups of project managers: One group worked at the systems development company, while the other worked for the Materials Administration of the Swedish defence. In the project, they acted as representatives for the *contractor* and *customer*, respectively.

5.4. The Usability Evaluator

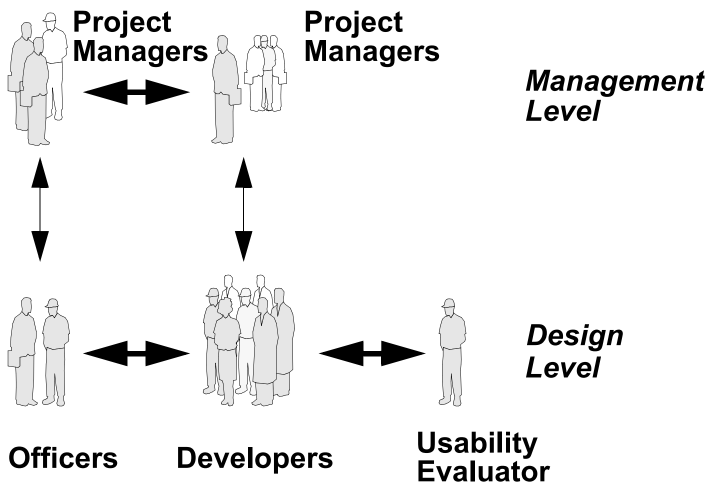
The role of the usability evaluator was to evaluate early design suggestions with respect to usability. Usability was seen as a quality which emerges when an application is used. The purpose of the evaluations was to provide formative feedback to the system developers, by highlighting aspects of design suggestions that may cause deficiencies in usability, and by participating in discussions regarding possible improvements of the application with respect to usability.

Working practices for formative usability evaluation include the use of empirical and analytical evaluation methods (Jeffries *et al.* 1991, Nielsen 1993). Empirical methods for formative usability evaluation make use of simulated use contexts, so that usability characteristics can be detected by user testing in an environment that is hopefully similar to the future use context. Analytical usability evaluation methods (e.g., heuristic evaluation, Nielsen 1993) make use of earlier research and experiences of common usability problems for predicting whether similar problems may occur for the actual application. The relation between common design options and generic human physical, perceptual and cognitive characteristics are supposed to be invariant enough to make it possible to use prior experiences as predictions for future usability problems.

5.5. Interaction Between Actors

The interaction between actors *inside each of the groups* was intensive and frequent. As far as I could observe, the interaction was also efficient, with few misunderstandings, and with use of terminology shared among the actors.

FIGURE 2. Groups of actors divided into two levels



The interaction between actors *in different groups* took place at two levels: The management level, and the design level (Figure 2). At the management level, the two project management groups discussed issues based on the contract. At the design level, the developers frequently discussed issues regarding domain issues with the officers, and issues regarding usability with the usability evaluator.

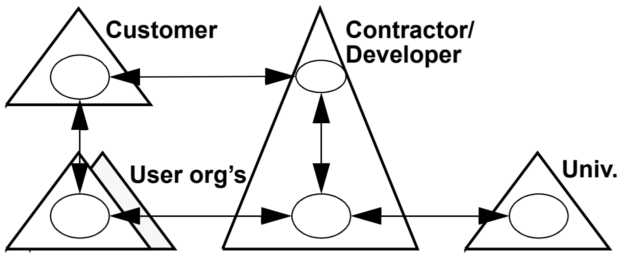
Interaction between the two levels was less frequent. The program project leader acted as a mediator between the group of developers and the project managers of the systems development company. The officers held meetings with the customer's project managers, in which they discussed issues regarding procurement. My understanding of these meetings is that the officers informed the customer's project managers about what was going on at the design level.

The borders between the five groups closely follow the border between different organisations. Figure 3 is a redrawing of figure 2 with triangles used to depict different organisations (cf. Grudin 1991a, Grønbæk *et al.* 1993).

6. Contextual Views of the Application

Interviews and discussions with the actors, as well as observations of their actions, revealed that the computer application was viewed *in context* by the actors. However, there was a great difference between the groups regarding *what* context they viewed the application in. This section tries to describe and illustrate these views.

FIGURE 3. Participating organisations



6.1. The Officers' View: An Improved Application in the Old Use Context

The officers had the currently existing application and its current use vivid in their minds. In particular, they could give detailed descriptions regarding *problems* which frequently occurred during use of this application.

On the basis of the observed problems, they often framed these problems as *demands for the new application*. These demands varied from slight adjustments of features existing in the current application to requirements for new technology. Often, these demands were expressed as *technical solutions* that they believed in. However, from the perspective of a professional system developer, these technical solutions were not very good. For many of them, it was obvious that they were made by laymen in systems development.

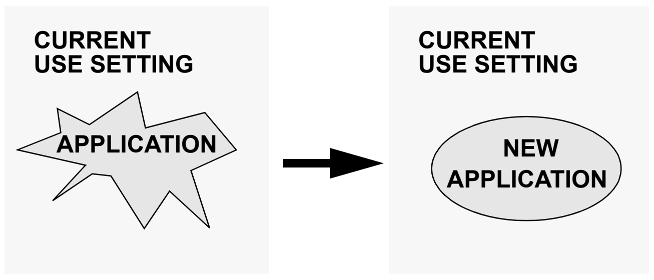
The officers were hopeful that new technology *in itself* would provide better usability. They frequently expressed confidence in greatly improved usability through the introduction of a graphical user interface and through technical solutions such as geographical information systems and high resolution colour presentation.

The officers had experience from using commercially sold computer applications for personal computers. These products had well-designed graphical interfaces, which the officers found easy to use. My belief is that the experience from these applications made the officers believe that graphical user interfaces inherently make computer applications easy to use.

While the awareness of existing problems in current use was high, I found clear indications that the officers had difficulties in detecting or envisioning other factors:

- They had difficulties in *articulating what worked well* in the current situation. It was easier for them to identify existing problems than existing strengths. Current practice was taken for granted.
- They had difficulties in seeing completely new options for support. They appeared to be trapped by their experience of the current application.
- Although they were able to generate suggestions for solutions to existing problems, they had very limited ability to estimate what *new* problems

FIGURE 4. The officers' view of the application in context



could occur if these new solutions were introduced.

To me, these aspects indicate a situation where the officers' experiences, visions and ideas for technical solutions should be used as input to the design process, but where their demands for the new application should not be seen as *The* solution. The officers lacked the technical competence and design experience to enable them to specify a relevant design.

The officers imagined an improved version of the existing application, in the current use setting. The envisioned improvements of the application were substantial with regard to usability, but their visions were not detailed. The officers frequently described that it should be possible to use the application "just by pointing and clicking." When asked to described how they envisioned particular services they often described very simple patterns of action, which lacked many actions that would be necessary. Their visions would not be possible to implement—their visions were simply not a finished design!

Figure 4 is a schematic illustration of the officers' view of the application in context. It has two parts, separating their experiences of the existing application and their vision for the new application. An imperfect application—with sharp edges used as illustrations of some irritating, and hence strikingly visible, aspects—in a well known use context is replaced by a much "smoother" application in an unchanged use context.[1] Almost everything is left unchanged, although some very irritating aspects of the current application are removed. The development process is not given much attention. It is primarily seen as a move from "now" to "the future," indicated in the figure by an arrow.

6.2. The Developers' View: An Evolving Computer Application Fitting into a Web of Constraints

The developers clearly described their effort to be concentrated in time. For them, the project started with the agreed contract and the requirements specification, and would end with an accepted delivery of the application. The subsequent use of the computer application was considered to be beyond the scope of their effort. The value of the project after delivery of the application was supposed to be the experiences and knowledge gained by the systems development company in the development process.

Similarly, the current application and the usability problems related to its use was to be considered out of the scope. These aspects should be reflected by the specified requirements, but were uninteresting *per se*.

Although the subsequent use of the computer application was considered beyond the scope of the project, the application's usability was still regarded as an important issue. Usability was seen as *a quality of the application*. Usability could thus be "built into" the application.

The development process was handled as a process where *constraints should be identified* and where *the emerging product should fit the identified constraints*.

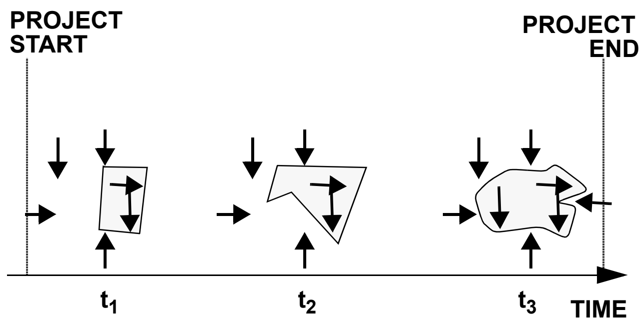
There were several sources for these constraints:

- The written requirements specification, which was part of the contract, was seen as an initial set of constraints.
- The officers—which by the developers were seen as 'the users'—were used as additional sources of constraints. They were frequently asked to specify in more detail what a statement in the written requirements specification "really" meant. They were also frequently used as decision makers, or 'acceptors', when the developers wanted additional constraints to be officially established.
- The tools or fourth generation languages which were used implied a large set of design constraints. These constraints can be divided into several levels: Certain constraints were imposed by the tool, and needed to be there if the tool should be used at all. Other constraints were possible to circumvent by adding modules programmed in low level languages. Still other constraints were imposed by templates and standards suggested by the tools, but possible to override. In all these cases, the developers seemed very eager to apply the identified constraints even when they could be circumvented.
- The identification and construction of templates, design rules and standards were given a great deal of attention already at an early stage. Typically, these new constraints were introduced without any major analysis of consequences or alternatives.
- In some instances, working habits that had been used by one of the developers were perceived by other developers to be preferred or decided ways of working. There was a tendency to copy others styles of working. The one developer who differed from this, by experimenting with and exploring alternatives until he found a solution that fitted his sense of good quality, received reprimands and complaints from other developers that his parts did not fit into the context.

The developers sought constraints for their work rather than attempting to reduce them. The developers were reluctant to challenge already identified constraints. This web of constraints became the context in which the evolving application was seen. The system developers planned their actions carefully so that the evolving computer application should fit its context.

Several explanations can be made for the importance given to these constraints:

FIGURE 5. The developers' view of the application in context



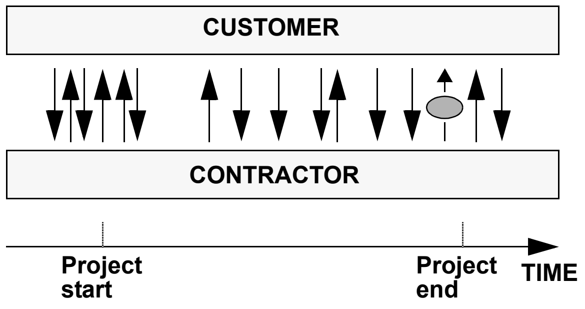
- The developers were eager to reduce the available design space. The less freedom developers have, the easier it is to coordinate the work of several developers. Furthermore, a heavily constrained design space facilitates discarding new and innovative ideas, which otherwise takes time to explore. This saves time, which was a scarce resource in the project.
- To follow constraints given by others implies a reduction of own risks, if the constraints are identified early. Design constraints which the developers are not allowed to violate must be detected early. If such constraints are not identified early enough, there is considerable risk that design decisions and subsequent development work have to be revised. However, if they are identified early enough, adopting them often means that the risk remains by those issuing the constraints.

Related to both these explanations is the observation that the developers often chose to be very "service minded," in the sense that they asked the officers what their wishes were, so that the developers could construct what was desired. However, when the officers asked for something that conflicted with the written requirement specification and would require extra development efforts, the developers chose not to follow the officers' wishes. Hence, the officers wishes were considered important when they could introduce additional design constraints, but not if they would violate or question already established constraints or require additional work.

Figure 5 is a schematic illustration of the context in which the developers viewed the application. The focus is clearly limited in time: The *project* was initiated with a contract, and ended with delivery and installation. Hence, both the use of the earlier application and the subsequent use of the new application was beyond the scope of the developers' attention. During the project, the application *evolved within a set of identified constraints*.

In the figure, arrows depict identified constraints. The web of constraints

FIGURE 6. The project managers' view of the application in context



makes up the context in which the application (grey) evolves over time. Three "snapshots" from this evolution are illustrated (t_1 to t_3). The "shape" of the application is designed to fit the constraints as well as possible. The view is limited in time to cover only the period in which the contractor is responsible for the evolution.

6.3. The Project Managers' View: A Business Agreement between Two Separate Parties

The two groups of project managers appeared to have a common view of application development: In the development project there are two main parties, acting as customer and contractor. Documents, goods and money are transferred between the two parties. When and how these transfers are made, as well as what is transferred, is regulated in advance. It is regulated in the contract, in standardized procedures, business traditions, as well as legally. It is assumed that these transfers are made in agreement. If disagreements should occur anyway, there are agreed ways of handling these conflicts smoothly.

Although I did not study these aspects in detail, I was surprised to note both how strongly formalized the exchanges were, and how strongly institutionalized the patterns of action appeared to be. Patterns of conduct and behaviour apparently were deeply rooted within this business tradition.

I was also somewhat surprised by the amount of exchange between the two parties. Although the most important events were the initial contract, the main deliverable, and the delivery acceptance and payment, a large number of other transfers of documents, goods, and money were also made. There were additional contracts, clarifications and redefinitions, subdeliveries, advance payments, documents of intermediate acceptance, protocols, etc.

The two parties maintained a strict separation. For example, there was considerable difference between external and internal documents (i.e., between public and company confidential documents). The division between externally spread and internal knowledge, opinions, motives, etc. was also strictly maintained.

FIGURE 7. The usability evaluator's view of the application in context

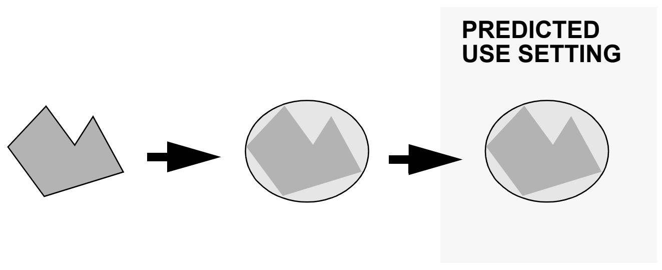


Figure 6 illustrates the view held by the two groups of project managers. Customer and contractor roles, and the exchange of goods between them, are in focus. The application (grey oval) is only one—although an important—object to be exchanged in the business process the application development was seen as. Hence, the application was seen in the context of strictly formalized business exchanges. An axis representing the temporal dimension (time) is included (but may be disregarded by a reader finding the illustration being too complex).

6.4. The Usability Evaluator's View: Predicting the FutureUsability of the Evolving Application

As indicated earlier, the usability evaluator was a somewhat different kind of actor in the studied development process. First, formative usability evaluations are not regularly used in the standard application development process at the systems development company. Instead, it was part of a research effort. Secondly, it must be noted that 'the usability evaluator' is the author of this chapter, although the format of the description is made similar to the format of the earlier descriptions.

Formative usability evaluation implies that the use qualities are assessed already at development time. Hence, the usability evaluator had to span a gap of time. The usability evaluator imagined each prototype or sketch in the context of its future use. A typical question to be answered by the usability evaluator was "If the ideas sketched in this early prototype were included in the delivered application, what usability characteristics would these parts of the application have during use?"

The usability evaluator thus worked with predictions. He predicted both the further development, and the future use. His task was to visualize an *existing* idea or part of the application in the context of the *future* application and its *future* use. Figure 7 illustrates the steps in this process. In the first part of the figure, the prototype is sketched with edges to illustrate its incompleteness. The second part illustrates how the usability evaluator predict the future application by adding parts and qualities missing in the prototype, but likely to exist in the final computer application. Finally, the imagined future application is placed into an imagined use context, where usability characteristics are forecasted. This last part of the figure shows the usability evaluator's primary focus.

7. Contextual Application of Expertise

All four groups of actors must be said to have seen the application *in context*. However, for each group, the context in which the application was seen was rather different. A common characteristic was that the context each of the actors used had a good fit to this actor's expertise. The contexts in which the application was seen *made it possible* for the actors to *apply* their expertise.

The two officers were experts of the application domain, and of the current use of the existing application. They had limited knowledge of, e.g., application design, project management, usability, and cognitive psychology. By viewing the new application in the context of use, they could *envision* the new application, but not design it. They were not able to understand the consequences of their envisioned design, neither for the developers nor for the ultimate users.

The developers viewed the application in the context of a web of constraints. This made it possible for them to apply their expertise: Knowledge of technical options, skills in combining technical options, and skills in assessing the options with respect to given constraints. By *interpreting* statements about user needs and the officers' wishes *as* constraints, they could use these as parts of a context in which they could apply their skills. Conversely, it helped them to regard experiences from use of the current application, and considerations of the use of the new application, to be beyond the scope of their effort. They assumed that experiences from current use and expectations for future use had already been converted to design constraints.

By looking at the application as a product (i.e., a deliverable), *the project managers* could apply their expertise in business procedures and legal matters. They could to a large degree isolate themselves from detailed design considerations. When design issues were brought to the project managers' attention, they were typically *seen as*, and *handled as*, deviations in agreed deliverables. Consequently, unanticipated problems for the developers could be handled as a prediction of delay in delivery, requiring renegotiations and possibly also economic compensation. Similarly, detection of unfortunate mistakes in the written requirements specification could be handled as an additional order from the customer to the contractor. Issues like these were handled in a (from the author's point of view) remarkably predefined and routinized way, hence smoothening up the handling of the issues (from the project managers' point of view).

The usability evaluator could apply his expertise if relevant information regarding the *intended use* of and *planned further development* was given for the parts which were to be evaluated. With such information, the usability evaluator could set up appropriate conditions for empirical evaluation and select suitable approaches for analytical evaluation, by building an imagined context of future use of the final application.

8. The Interrelation of Contextual Views

The contexts in which each group of actors visualized the application provided good opportunities for them to apply their own expertise. However, a drawback with separate views is that the differences may create obstacles for communication between actors. This section of the chapter is devoted to the problems which emerged when the various contextual views for the application conflicted.

8.1. Management Level View Imposed on the Design Level

The project managers' view, characterized by a clear division between procuring organisations and delivering organisations, was enforced upon developers. This created a situation in which the developers and the officers had partly different information, but where they often concluded that they were not allowed to inform "someone from the other side." The officers, in particular, found this situation quite annoying.

In cases where the developers and the officers did not completely agree on the interpretation of the written requirements, the issue was frequently "sent up" to the management level instead of being solved between the developers and the officers. At the management level, the issues were resolved in a formal manner. Hence, design issues were *reinterpreted* as legal issues.

I was told by some developers that this practice did not always work well in practice. In other projects, they had experienced that the two groups of project managers finally agreed upon a compromise which the developers found to be an extremely odd solution to the actual problem. In such cases, they felt that the management level lost the needed sense of why the issue was so intesively discussed at the design level.

Promises to deliver intermediate deliverables sometimes clearly interfered with what the developers thought was the best way of working. Although it was the intention at the managerial level that these deliverables would be intermediate results from the development process, the developers described that they had to do certain time-consuming activities in order to create the deliverables, rather than doing what they needed to do for the progress of the systems development process.

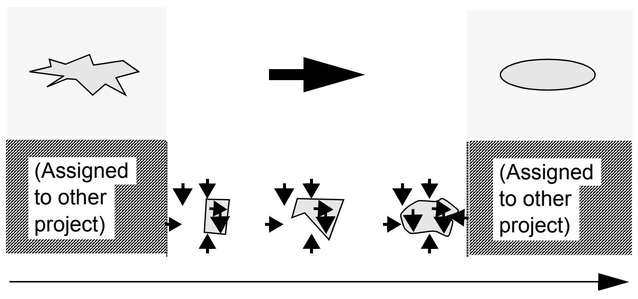
8.2. The Officers and the Developers

The developers frequently used the officers as interpreters of the written requirements specification. Frequent questions were "What do you mean with this statement?" and "How do you want this feature to work?" This can be seen as a situation where the developers were extremely service minded. It can also be seen as a situation where the officers were forced to make difficult design decisions in order to constrain the design space for the developers.

The officers complained that they did not get enough opportunities to explain their visions for the new application. They found themselves in a situation where they were forced to respond to detailed questions, but where the sum of the detailed answers did not measure up to their vision.

The developers complained that the officers had too many visions that were not contained in the contract. They told me that when they asked the officers to explain what was *meant* by a particular

FIGURE 8. Combination of the officers' and the developers' view of the application in context



statement in the written requirement specification, they often received an answer that was rather a request for *something* more. In particular, the program project leader often told me sighing that it was extremely difficult to restrain the officers' wishes and expectations. Her view was that the officers tried to extend the content of the agreed contract without having to pay more. Hence, she felt that a task of the developers should be to avoid new ideas which would extend the project.

When the application gradually began to emerge, the officers became somewhat surprised with what they saw. It did not conform to the vision they had. One officer explained that he found the emerging application more awkward than he had expected. This view may partly have been caused by his difficulties in reading and assessing early prototypes and sketches made by the developers, and partly by the fact that the officers had an idealised vision of the new application.

The officers frequently blamed themselves for not having realized important requirements in due time. They expressed that this was their fault. They demanded the developers to be skilled in systems design and realisation, but regarded themselves responsible for expressing the requirements.

There are clear differences between the officers' and the developer's contextual views of the application (figures 4 and 5). The major difference deals with the time dimension. While the officers' main focus was on the time they used the existing application, and when they will use the new application, both these periods were beyond the developers' scope. Another main difference is that statements the officers primarily regarded to express needed improvements of the existing application, were by the developers seen as design constraints.

Figure 8 combines figures 4 and 5 in order to illustrate how little overlap there is in the time dimension. The upper part illustrates the officers' view, while the lower part illustrates the developers' view.

The officers had wishes for the new application, but were not able to express them as a complete set of definite requirements for the application. The developers used the officers' stated requirements as constraints for the development without challenging them. Both parties *assumed* that stated requirements should be the basis for their communication, but since they did not understand each other's viewpoints, they were not able to detect the uncertainties concealed in the specified requirements. Instead of working cooperatively in challenging the initial ideas and performing a cooperative search for what a good application should be (cf. Ehn 1988, Greenbaum & Kyng 1991, Mogensen 1992 and 1994, Schuler & Namioka 1993), the officers blamed themselves for not being able to identify all requirements, while the developers struggled to find technical solutions in a web of constraints which they regarded to be fixed.

8.3. The Developers and the Usability Evaluator

The usability evaluator delivered several large evaluation reports, with a great deal of warnings regarding possible future usability defects in use of the future application. Many of these warnings did not result in immediate design changes.

Thus, an interesting question to pose is *why* warnings regarding future defects of an application did not lead to design changes. The differences in views between the usability evaluator and the developers can explain a substantial part of this reluctance. Many of them were also confirmed in later interviews and discussions with developers.

The evaluation reports often challenged constraints considered by the developers to be fixed. When shortcomings were identified by the evaluator, typical responses from the developers were "But we already have an agreement with the users that...", "But this would not conform to the specified requirements!" and "But this would require a lot of additional work not planned for!" Although usability was initially stated to be an important objective, identified constraints were in practice far more important for the developers than warnings about possible future usability defects.

As stated earlier, the developers adopted the view that the project ended with the delivery and formal acceptance of the application. The future use of the application was considered beyond the scope of their effort. This view was detrimental for the usability evaluator, who *motivated* his requests for design changes by pointing out possible usability problems *in future use*.

Many findings of the usability evaluations were highly appreciated by the developers, however. In retrospect, it can be seen that evaluation findings that could help the developers to reduce the available design space were those that made the most impact. A developer that is in the process of making a choice between two alternatives is very susceptible to arguments about positive and negative effects of each of these alternatives.

For some aspects, the developers even prompted the usability evaluator to make the necessary decisions. To just take a single example as an illustration, the developers often wanted the usability evaluator to make normative statements about what shade of colour would be "the most usable" for a particular purpose.[2]

From the evaluator's point of view, it was not the most important findings about possible usability defects that led to design changes. Instead, many minor remarks had considerable impact, while several extremely important warnings about future shortcomings appeared to be ignored.

The evaluator also encountered problems with what he regarded as a lack of plans for what the total application would be like. The evaluator wanted these plans for understanding how a designed part of the system or a prototype related to the total system. The developers did not see a corresponding need to make such a plan. For them, the total system was simply the sum of all parts, where each part was designed in sequence. Thus, the totality would be visible first towards the end of the project.

From the developers' point of view, the usability evaluator often asked for design alternatives which questioned already established constraints. When there was already an agreement with the officers or the customer, the usability evaluator could question that agreement. To follow the usability evaluator's advice would be a definite hindrance to the progress of the development work.

8.4. The Officers and the Usability Evaluator

The officers and the usability evaluator had complementary ways of looking at the application. While the officers had better knowledge of the application domain, the usability evaluator had considerably more experience in "reading" prototypes and sketches, and in making conclusions about possible future consequences of design alternatives.

The officers and the usability evaluator soon found a common interest in their exchange of information. However, this exchange of information soon had to stop. The program project leader found negative effects in the cooperation between the officers and the usability evaluator. Since the developers already had problems in keeping the officers' wishes within the limits of the contract, it was considered a disadvantage to establish cooperation between them and the usability evaluator (who repeatedly asked for design alternatives beyond the established constraints). In addition, the program project leader felt that there were often competing objectives between the 'customer side' and the 'contractor side.' She wanted the usability evaluator to help *the developers* to make appropriate decisions. This was also the role agreed upon for the usability evaluator in the project.

Other problematic aspects with officer/evaluator cooperation were identified, but not further discussed since the cooperation ceased. These aspects included the fact that much of the information in the project was internal (i.e., available for only *one* of the two parties customer/contractor) and thus could not be used in cooperative design discussions, and that the use of the officers was expensive for the project. The participation of the officers in the project was regulated in the contract and additional use of them for the project could be costly. In such a situation, cooperation between the officers and the usability evaluator could have introduced hindrances for the developers or enforced a renegotiation of the contract.

9. Conclusions

The case study has identified how the actors in the systems development project all held a contextual view of the computer application, but that the context they placed the application in varied considerably among the groups of actors.

9.1. *Contexts for Application of Expertise*

Each of the views made it possible for the actors to apply their core expertise. The developers could focus on finding technical solutions that fitted well into the identified constraints. The project managers could apply their knowledge on business and legal matters within their view of the development process as an exchange of goods. The officers could discuss their visions for the new computer application in the context of their experiences from use of the existing application. The usability evaluator, finally, could apply his skills of usability evaluation within an imagined future context of use.

Hence, the differences in views are important. We need to understand that we should not strive for one, single context to view the emerging computer application in. Such a single view would risk that some actors would not be able to apply their expertise efficiently and effectively.

9.2. *Communication and Cooperation*

A major problem with the differences in views was that the differences were not identified within the development project. Each group of actors assumed that the other actors had rather similar views as themselves. This led to misunderstandings, and other communication problems.

Although the different views are important, we must learn how to bridge the differences. Each actor must understand enough of the different contexts for being able to communicate with the other actors without misunderstandings.

In order to facilitate communication between domain experts (”users”, etc.) and technical experts (”developers”, etc.), it is crucial that *at least* the actors in one of the groups understand the relationship between *application use* and *design constraints*, and are able to translate between them. If the actors in only one of the groups understand it, a burden is placed upon these actors; they become responsible for detecting the communication flaws.

9.3. *Different Views of Quality*

The different contexts implied different views on quality. For the system developers, high quality primarily meant the delivery of a computer application which fitted within the identified constraints, and hence also was developed within the given limits of time and cost. Usability was identified as an important characteristic, but seen as a quality which should be assessed at time of delivery. Subsequent use of the computer application was out of the developers’ scope.

For the project managers, high quality primarily meant a smooth exchange of goods between customer and contractor, according to the plan. A delay in delivery was in effect seen as a quality problem in itself (rather than as an effect of another, underlying, quality problem).

For both the officers and the usability evaluator, characteristics of computer application use was in focus for quality judgements. Their quality views diverged, however: The officers’ view was primarily based upon their prior experiences of use and their visions of future use. The usability evaluator primarily worked with predictions based upon prototypes and sketches produced within the development process.

Such differences in views may exist within a systems development project. For not being detrimental to the process, they must be in harmony with each other, however. They are in harmony of each other, if an identified drawback with the design is simultaneously—by different actors—seen as a usability problem, a deviation from user expectations, a violation of a given constraint, and a reason for not delivering or accepting the application as it is. If, however, a particular aspect of the design is seen as high quality within one view, but low quality within another view, there is a problem with conflicting views on quality.

9.4. Pressure to Act Within a Particular View

In three of the four groups, it was easy to identify how the actors felt a pressure to keep their actions within the limits given by their particular view.

The developers were eager not to violate identified constraints. Hence, they had problems to accommodate to usability findings reported by the usability evaluator, and to new ideas from the officers. In cases where these new findings and ideas violated identified constraints, the new findings and ideas were rejected.

The evaluator did not take stated requirements and stated restrictions for granted. Instead, he tried to find what would actually work (or not work) in its use context, regardless of whether this was in accordance to or contrary to what was stated in the contract or wished by the officers. Hence, he felt a pressure to go beyond what was stated in the contract.

The managers maintained the separation between the customer and the contractor. If some actions would be done without a contract, or outside what was stated in the contract, several difficult issues would emerge: Who will pay for this work? Who will be the owner of the information/knowledge acquired? Is there a risk that our company loses a competitive advantage? Who will be responsible if anything goes wrong?

The officers were an exception. Instead of keeping within the realm of their experiences and visions, they often phrased themselves in the form of technical requirements. It was obvious, however, that this transition to another domain than their own resulted in loss of higly relevant information.

9.5. Being Trapped Within a Particular View

Several of the actors also indicated that they sometimes felt trapped within the established views. When discussing ideas from participatory design with the developers, a common comment was that "this appears valuable, but it cannot be used here."

When discussing the possibilities for more flexible forms of contracts (such as in Thomsen 1993, and case 2 in Grønbæk *et al.* 1993) with the project managers at the development company, the response was that "we would also like to have such flexible contracts, but our customer won't."

9.6. The Lack of a Designer Role

A particularly interesting effect of the combination of views in the project is the lack of a designer role that emerged. The developers assumed that they could identify and establish constraints for the design, and make a technical solution based on that. In a sense, the developers assumed that *someone else* had determined what the application should do. The officers assumed that the developers should make something good based on the visions the officers gave them. Thus, they assumed that *someone else* should design the behaviour of the application. This situation was not clearly visible, neither for the officers nor for the developers.

It is an open question is in which way a more clearly defined designer role would have been beneficial for the project or not. There is considerable risk that such a designer would have found her work constrained by the views of other actors.

10. A Programme for Improvements

The research described here is primarily descriptive and interpretative. It is difficult to extract normative recommendations on basis of these findings. It is possible, however, to point at some interrelations between the findings reported here, and suggestions made by other researchers in our field—not at least in the other chapters of this book.

First and foremost, however: There are no simple solutions. Systems development is extremely difficult, and each situation is new and specific (Andersen *et al.* 1990). Rather than looking for *the* solution, we should strive for successive improvements. This section outlines some promising attempts in the research community which can provide a basis for such improvements.

10.1. Cooperation and Envisionment in Design

Much work has been made in the participatory design community regarding cooperation and envisionment in design (e.g, Ehn 1988, Bødker & Grønbæk 1991, Greenbaum & Kyng 1991, Schuler & Namioka 1993, and Grønbæk *et al.* 1997, Bratteteig & Stolterman 1997).

This is clearly an important strain of work for achieving cooperation and envisionment in design. The findings in the research presented in this chapter can be used to highlight some important aspects to pay attention to in further research:

- Issues at the managerial level appear to have more impact in professional systems development than in systems development performed in action research. Issues of time, cost, contracts, security, competition, and successive replacement of personnel in the project group, have a considerable impact on professional systems development. It is not obvious how to make participatory design approaches applicable in the actual setting of professional systems development. Blomberg *et al.* (1997) provide examples of these difficulties in a product development context.
- In professional systems development, design issues are tightly connected to the overall mission of actually delivering a computer application which will be used for a long time. We cannot focus only on design, but rather on the interplay between analysis, design, realisation and use. Grønbæk *et al.* (1997) is an important example of what is needed in this vein of work.

- We must learn to understand and handle the relation between at one hand different views and at the other hand conflicting interests among stakeholders. In the project discussed in this chapter, differences in interests identified at the management level appears to have blocked a confrontation of the actors' different views. The participatory design community appears to have drifted from a focus on conflicting interests to a focus on different but harmonious views, but without really having found a way to handle both these issues simultaneously (cf. Bjerknes & Bratteteig 1995). McMaster *et al.* (1997) address these issues further.
- The apparent lack of a designer role in the studied project indicate a situation almost totally contrary to the ideal of creative and visionary design, as described by Bratteteig & Stolterman (1997). We need further research on how to achieve creativity in professional systems development (cf. Mathiassen 1988; Stolterman 1992).

10.2. Achieving Harmony among Different Views of Quality

As discussed in the conclusions, it is important that different views of quality held within the project are in harmony with each other (cf. Vidgen *et al.* 1993). The major obstacle to this seems to be contracts which does not take usability issues into account. The issue is difficult, but of utmost importance. Work on usability engineering (Whiteside *et al.* 1988; Carlshamre 1994a, 1994b) and on more flexible forms for contracts (Thomsen 1993, cf. Grønbæk *et al.* 1993) are important steps along this way, but much more work is needed.

10.3. Finding Languages for Talking about Qualities

The framework sketched by Ehn *et al.* (1997) provides an interesting illumination of the tension between the different views held by the actors and the language used for communication about the application under development. The view held by the officers, rooted in their experience with the existing application, maps rather well to Ehn's *et al.* aesthetic perspective ("Form"). The usability evaluator's view was also concerned with prediction of the work-oriented interplay between users and the computer application, i.e. an ethical perspective ("Function"). The views held by the managers and the developers were primarily structural, although they differed regarding to the objects which were being discussed (technical issues vs. legal issues). The language used for expressing these different views were primarily structure oriented, however. This is most easily identified for the officers, who tried to phrase their *experiences* ("Form") in terms of *requirements* for the new application ("Structure"). New ways of talking about form and function may make help in making differences in views among actors clearer. With only a language based upon a structural view, differences in views may be hidden.

10.4. Bridging the Research Traditions in SystemsDevelopment

Research on systems development is fragmented into several traditions, with surprisingly little overlap. Many of the issues identified in the studied case span over several of these traditions, however.

Although we have begun to understand how to carry out usability evaluations (HCI research), how to cooperate between users and developers (participatory design research) and how to handle development contracts between customers and contractors (software engineering research), we lack the knowledge of how to combine this knowledge in large projects.

Notes

[1]Throughout the figures in this article, an application illustrated without edges is used to denote an application that in some sense is “better” than an application with edges. This hopefully have some intuitive appeal, even if it is left somewhat undefined what “better” means in different circumstances.

[2]This was a question which the usability evaluator both had considerable difficulties to answer, and which he often regarded to be of minor importance.

Acknowledgements

A very large number of people have contributed to the work reported here. First and foremost, I would like to thank the different actors in the systems development project. NUTEK, the Swedish National Board for Industrial and Technical Development, financed the study through their information systems frame programme. The European COST-14 (Cooperation Technology) project IMPACT provided further funding. The members of the Usability Matters group at Linköping University, as well as reviewers, discussants and participants of the 1995 IRIS and Computers in Context conferences, provided valuable comments and questions about this work and the presentation of it. Finally, Preben Mogensen, Kristin Braa and the editors helped to improve the text for the book.

References

Andersen, N. E., Kensing, F., Lundin, J., Mathiassen, L., Munk-Madsen, A., Rasbech, M. & Sørgaard, P. (1990). *Professional systems development.* Prentice-Hall, New York.

Bjerknes, G. & Bratteteig, T. (1995). User participation and democracy: A discussion of Scandinavian research on systems development. *Scandinavian Journal of Information Systems,* 7(1):73-97.

Blomberg, J., Giacomi, J., Mosher, A. & Swenton-Wall, P. (1993). Ethnographic field methods and their relation to design. In Schuler & Namioka (1993), Pages 123-155.

Blomberg, J., L. Suchman & R. Trigg, (1997). Back to Work: Renewing Old Agendas for Cooperative Design. In (Kyng & Mathiassen 1997), Chapter 10.

Bødker, S. & Greenbaum, J. (1988). *A feeling for systems development work: Design of the ROSA project.* Report DAIMI PB-246, Computer Science Dept., Aarhus University.

Bødker, S. & Grønbæk, K. (1991). Cooperative prototyping: Users and designers in mutual activity. *International Journal of Man-Machine Studies,* 34: 453-478.

Braa, K. & R. Vidgen (1997). Action case: exploring the Middle Kingdom in information systems research. In (Kyng & Mathiassen 1997), Chapter 14.

Bratteteig, T. & E. Stolterman (1997). Design in groups—and all that jazz. In (Kyng & Mathiassen 1997), Chapter 11.

Carlshamre, P., (1994a). *A collaborative approach to usability engineering: Technical communicators and system developers in usability-oriented systems development.* Linköping Studies in Science and Technology, Licentiate Thesis 455.

Linköping University, Linköping, Sweden.

Carlshamre, P., (1994b). Technical communicators and systems developers collaborating in usability-oriented systems development: A case study. *Proceedings of SIGDOC'94*. ACM, New York.

Ehn, P., (1988):*Work-oriented design of computer artifacts*. Arbetslivscentrum, Stockholm.

Ehn, P., T. Meggerle, O. Steen & M. Swedemar, (1997). What kind of car is this sales support system?. In (Kyng & Mathiassen 1997), Chapter 5.

Greenbaum, J. & Kyng, M., editors, (1991). *Design at work: Cooperative design of computer systems*. Lawrence Erlbaum, Hillsdale N.J.

Grønbæk, K., Grudin, J., Bødker, S. & Bannon, L. (1993). Achieving cooperative systems design: Shifting from a product to a process focus. In Schuler & Namioka (1993), 79-97.

Grønbæk, K., M. Kyng & P. Mogensen, (1997). Cooperative experimental system development—cooperative techniques beyond initial design and analysis. In (Kyng & Mathiassen 1997), Chapter 8.

Grudin, J. (1991a). Interactive systems: Bridging the gaps between developers and users. *IEEE Computer*, April, 59-69.

Grudin, J. (1991b). Systematic sources of suboptimal interface design in large product development organizations. *Human-Computer Interaction*, 6(2).

Gummesson, E. (1988). *Qualitative methods in management research*. Studentlitteratur, Lund, and Chartwell-Bratt, Bromley.

Jeffries, R., Miller, J., Wharton, C. & Uyeda, K. (1991): User interface evaluation in the real world: A comparison of four techniques. In *Human Factors in Computing Systems (CHI'91 Proceedings)*. ACM, New York.

Kyng, M. & L. Mathiassen, editors, (1997). Tentatively titled: *Computers in Context: Joining Forces in Design*. MIT Press, Boston.

Mathiassen, L. (1988). Kreativitet og disciplin i system design. *DATA Nordisk Datanytt* 18(6):33-37. (In Danish).

McMaster, T., M. C. Jones & A. T. Wood-Harper, (1997). Impementation planning: a role for "information strategy". In (Kyng & Mathiassen 1997), Chapter 9.

Mogensen, P. (1994): *Challenging practice: An approach to cooperative analysis*. Thesis. DAIMI PB-465. Computer Science Dept., Aarhus University, Århus, Denmark.

Mogensen, P. (1992): Towards a provotyping approach in systems development. *Scandinavian Journal of Information Systems*, 4:31-53.

Nielsen, J. (1993): *Usability Engineering*. Academic Press, Boston.

Schuler, D. & Namioka, A., editors, (1993): *Participatory design: Principles and practices*. Lawrence Erlbaum, Hillsdale, N.J.

Stolterman, E., (1992): How system designers think about design and methods: Some reflections based on an interview study. *Scandinavian Journal of Information Systems,* 4:137-150.

Thomsen, K. S. (1993): The Mentor project model: A model for experimental development of contract software. *Scandinavian Journal of Information Systems*, 5:113-131.

Vidgen, R., Wood-Harper, A. T. & Wood, R. (1993). A soft systems approach to information systems quality. *Scandinavian Journal of Information Systems*, 5:97-112.

Whiteside, J., Bennett, J. & Holtzblatt, K. (1988). Usability Engineering: Our experience and evolution. In Helander, M., editor. *Handbook of human-computer interaction*. Elsevier (North-Holland), Amsterdam. Pages 791-817.

Wiklund, M., editor, (1994). *Usability in practice: How companies develop user-friendly products*. AP Professional, Boston.